AFRL-VA-WP-TR-2004-3014

HIGHER ORDER COMMON PLATFORM FOR COMPLEX MULTI-PHYSICS COMPUTATION

Dale K. Ota, S.V. Ramakrishnan, Vijaya Shankar, and Ramakanth Munipalli

HyPerComp, Inc.
31255 Cedar Valley Drive, Suite 327
Westlake Village, CA 91362

MARCH 2003

Final Report for 05 June 2002 – 05 March 2003

THIS IS A SMALL BUSINESS INNOVATION RESEARCH (SBIR) PHASE I REPORT

STINFO FINAL REPORT

AIR VEHICLES DIRECTORATE
AIR FORCE MATERIEL COMMAND
AIR FORCE RESEARCH LABORATORY
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

NOTICE

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION: HIGHER ORDER COMMON PLATFORM FOR COMPLEX MULTI-PHYSICS COMPUTATION.

/s/
Datta V. Gaitonde
Leader
High Speed Computational Research Team
Computational Sciences Branch

/s/
Datta V. Gaitonde
Project Engineer
High Speed Computational Research Team
Computational Sciences Branch

/s/
Douglas C. Blake
Branch Chief
Computational Sciences Branch
Air Vehicles Directorate

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YY)* March 2003	**2. REPORT TYPE** Final	**3. DATES COVERED *(From - To)*** 06/05/2002 – 03/05/2003

4. TITLE AND SUBTITLE HIGHER ORDER COMMON PLATFORM FOR COMPLEX MULTI-PHYSICS COMPUTATION	**5a. CONTRACT NUMBER** F33615-02-M-3224
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 65502F
6. AUTHOR(S) Dale K. Ota, S.V. Ramakrishnan, Vijaya Shankar, and Ramakanth Munipalli	**5d. PROJECT NUMBER** 3005
	5e. TASK NUMBER 40
	5f. WORK UNIT NUMBER LK
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) HyPerComp, Inc. 31255 Cedar Valley Drive, Suite 327 Westlake Village, CA 91362	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Vehicles Directorate Air Force Research Laboratory Air Force Materiel Command Wright-Patterson Air Force Base, OH 45433-7542	**10. SPONSORING/MONITORING AGENCY ACRONYM(S)** AFRL/VAAC
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S) AFRL-VA-WP-TR-2004-3014

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
This is a Small Business Innovation Research (SBIR) Phase I report. Report contains color.

14. ABSTRACT
The objective of Phase-I of this project was to demonstrate feasibility of exploiting the commonality in the structure of the governing equations in fluid dynamics and electromagnetics to develop a higher order Magneto-Gas-Dynamics (MGD) solver, based on the 3D, unstructured, parallel discontinuous Galerkin (DG) CEM (Computational ElectroMagnetics) code TEMPUS under development at HyPerComp. This objective was met by demonstrating conversion of the TEMPUS code, to an inviscid flow solver with minimal modifications. The only required changes to the code were the incorporation of appropriate flux computations and boundary conditions, while the structure of the code remained essentially same. This code conversion was preceded by the development of an implicit scheme based on a novel linearization technique for the flux terms, and the application of the DG scheme to classical problems in 1D and quasi 1D. The 1D demonstration included extension of the DG formulation to MGD equations, and successful application of the scheme to the Brio-Wu shock-tube problem and Wu's intermediate shock problem. It has been concluded that a strong platform for high order multi-physics computations may be built on the code structure and formulation developed in this activity, with extensions to elliptic and parabolic differential equation systems.

15. SUBJECT TERMS
SBIR report, Discontinuous Galerkin, Higher order, multi-physics, magnetohydrodynamics

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT:**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** (Monitor)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. Datta Gaitonde
Unclassified	Unclassified	Unclassified	SAR	50	**19b. TELEPHONE NUMBER** *(Include Area Code)* (937) 904-4031

Table of Contents

Higher-Order Common Platform for Complex Multi-Physics Computation
Final Report

1.0 Introduction

The objective of Phase 1 of this project was to demonstrate feasibility of exploiting the commonality in the structure of the governing equations in fluid dynamics and electromagnetics to develop a higher-order Magneto Gas Dynamic (MGD) solver based on the 3D, discontinuous Galerkin (DG), CEM (Computational ElectroMagnetics) code TEMPUS [1] under development at HyPerComp. The MGD solver was to be restricted to inviscid flows. With a view to achieving this objective, we first investigated implementation of a DG scheme for solving the one-dimensional (1D) Euler equations, the governing equations for the flow of an inviscid fluid in 1D. The Euler equations represent one of the simplest non-linear set of partial differential equations (PDE) of relevance to fluid mechanics and wave propagation. We studied application of the DG algorithm for numerical solution of 1D Euler equations as applied to simulation of flow in a shock tube, flow through a quasi-1D convergent-divergent nozzle, and propagation of a periodic density wave. The implementation employed was based on a novel linearization technique for the flux terms that simplified the development of explicit schemes as well as a point-implicit scheme. Based on the linearization, an implicit formulation for the 1D Euler equations was developed. The DG formulation tested both monomials and orthogonal polynomials as basis functions. The formulation was extended for inviscid, 1D, MGD equations and solutions were obtained for the Brio-Wu shock-tube problem [2] and the Wu intermediate shock problem [3]. This was followed by the conversion of the 3D DG CEM code, TEMPUS, to a perfect gas Euler solver. The experience gained from the 1D studies enabled efficient code conversion. Inviscid flow over a 2D cylinder was used as the test problem to verify the veracity of the conversion. Future work will add the additional equations required to complete the MGD equation set. The code conversion task validated our initial assumption that the commonality in the structure of the governing equations cast in a conservation form in many disciplines such as CEM and CFD may be exploited in developing a common platform for solving problems in multi-disciplines. Design strategies for the development of a common platform for multi-physics computation were also investigated.

In the following sections we present the details of our work in Phase 1. The fundamentals of a DG scheme are described in section 2. Development of a DG scheme using orthogonal polynomials for a system of conservation laws in 1D is detailed in section 3. Numerical schemes developed using the DG scheme for simulation of inviscid, 1D and quasi-1D flows are presented in section 4. Results obtained using the DG scheme for some classical problems are discussed in section 5. An implicit formulation of the DG scheme is presented in section 6. The 3D DG Euler equations are described in section 7. Evaluation of some integrals involved in the 3D formulation is discussed in section 8. Steps involved in the conversion of the CEM code to an Euler solver are detailed in section 9. Some 3D test results are shown in section 10. Design of a common platform for multi-physics computation is discussed in section 11. Suggestions for the extension of the work done under Phase I are presented in section 12.

2.0 Discontinuous Galerkin Algorithm

In this section, a brief description of the discontinuous Galerkin algorithm is presented by applying it to the following generic system of conservation laws:

$$\frac{\partial \vec{Q}}{\partial t} + \nabla \cdot \vec{F}(\vec{Q}) = \vec{S}(x,y,z,t) \tag{2.1}$$

where $\vec{Q}$ is the vector of conserved quantities, $\vec{F}$ is the flux tensor, and $\vec{S}$ is the vector of source terms.

The computational domain (D) is partitioned into Ne non-overlapping subdomains (elements) De. Within each element, the solution $\vec{Q}$ is projected onto a local basis vector $v_e^i(x,y,z)$; e=0,…,Ne-1 ($e \rightarrow$ element), and i=0,1,…,n-1. It is represented by

$$\vec{Q}(x,y,z,t) = \sum_{i=0}^{n-1} \vec{Q}_e^i(t) v_e^i(x,y,z), \quad \text{for } (x,y,z)\ \varepsilon\ De \tag{2.2}$$

where $\vec{Q}_e^i(t)$ are time-dependent coefficients. The 'Discontinuous' in 'Discontinuous Galerkin' refers to the property of eqn.(2.2) by which the approximation for conserved quantities, while continuous within an element, is not necessarily continuous across neighboring elements.

The unknown coefficients $\vec{Q}_e^i(t)$ are computed via a weak formulation of eqn (2.1), obtained by multiplying it with each of the basis functions $v_e^i(x,y,z)$ and then integrating by parts over each element De to obtain

$$\int_{De} \left[v_e^i \frac{\partial \vec{Q}}{\partial t} - \nabla v_e^i \cdot \vec{F} \right] dV + \int_{\partial De} \left[v_e^i \vec{F} \right] \cdot d\vec{A} = \int_{De} \vec{S} v_e^i dV; \quad \text{e=0,…,Ne-1,} \quad \text{i=1,2,…,n-1} \tag{2.3}$$

where ∂De is the boundary of element De. The 'Galerkin' in 'Discontinuous Galerkin' refers to satisfying a weak form of eqn.(2.1) rather than eqn.(2.1) itself..

The volume and boundary integrals in eqn.(2.3) may be evaluated using either numerical quadratures or exact integrals depending on the basis functions employed. In either case the boundary integral involves evaluation of the flux tensor $\vec{F}$ at points on the boundary. Since $\vec{Q}$ is not necessarily continuous across an element boundary, we need a procedure for computing a unique value for $\vec{F}$ at a boundary point. While $\vec{F}$ can be evaluated in a number of ways, computing it by solving the Riemann problem based on the 'left' and 'right' states for $\vec{Q}$ computed at an element interface using eqn.(2.2) for the 'left' and 'right' elements respectively has produced the highest phase accuracy in our numerical experiments. Solving the Riemann problem on the boundary of an element is physically equivalent to computing $\vec{F}$ by considering only those waves, which are propagating towards the boundary, and ignoring those which are propagating away from it.

One of the most popular Riemann solvers employed in CFD (computational fluid dynamics) is the approximate Riemann solver developed by Roe [4]. Computation of $\vec{F}$ at a boundary point (BP) using Roe's Riemann solver starts by computing $\vec{Q}$ at the point from eqn.(2.2) for the two parent elements of the boundary point. Let these $\vec{Q}$ values be denoted by $\vec{Q}_L$ and $\vec{Q}_R$ where the subscripts *L* and *R* refer to the "Left" and "Right" element respectively. The flux tensor at the boundary point may now be written as

$$\vec{F}(BP) = \frac{1}{2}\left[\vec{F}\left(\vec{Q}_L\right) + \vec{F}\left(\vec{Q}_R\right)\right] - \frac{1}{2}|J|\left(\vec{Q}_R - \vec{Q}_L\right) \tag{2.4}$$

$|J|$ is given by

$$|\mathrm{J}| = \vec{R}\left|\vec{\Lambda}\right|\vec{L} \tag{2.5}$$

where $\vec{R}$ and $\vec{L}$ are the "right" and "left" eigen vectors of the Jacobian $J = \frac{\partial \vec{F}}{\partial \vec{Q}}$ and $\left|\vec{\Lambda}\right|$ is a diagonal matrix whose elements are the absolute values of the eigen values of *J*. Equation (2.4) may be interpreted as computing the flux vector at a boundary point using the central difference operator, $\frac{1}{2}\left[\vec{F}\left(\vec{Q}_L\right) + \vec{F}\left(\vec{Q}_R\right)\right]$ and the dissipation term $\frac{1}{2}|J|\left(\vec{Q}_R - \vec{Q}_L\right)$. Computation of the dissipation term involves evaluation of the eigen values and eigen vectors of the Jacobian matrix. In order to avoid expensive evaluation of eigen vectors without severely compromising the accuracy of the scheme, we employ a simpler form of eqn. (2.4) wherein the dissipation term is computed as $\frac{1}{2}|\lambda_{\max}|\left(\vec{Q}_R - \vec{Q}_L\right)$. This leads to the following expression for the flux vector at a point on an element boundary:

$$\vec{F}(BP) = \frac{1}{2}\left[\vec{F}\left(\vec{Q}_L\right) + \vec{F}\left(\vec{Q}_R\right)\right] - \frac{1}{2}|\lambda_{\max}|\left(\vec{Q}_R - \vec{Q}_L\right) \tag{2.6}$$

We refer to such a scheme as the "scalar dissipation" scheme because the computation of the dissipation term involves only a scalar coefficient and not a vector coefficient as in the case of Roe's Riemann solver.

The choice of the basis functions $v_e^i(x,y,z)$; i=0,1,…,n-1 determines a particular type and accuracy of the discontinuous Galerkin method. One can choose monomials such as 1, x, y, z, x^2, y^2, etc. or other functions such as orthogonal polynomials as basis functions. Obviously, with a larger basis function set, one can approximate the solution more accurately within each element, but at the expense of higher computational cost. We refer to the method as DG-pn, where n is the order of the polynomial representing the solution. It has been shown in the literature that a DG-pn scheme will be of order at least (n+1), and of order up to (n+2) in certain cases.

A second order TVD scheme also approximates the variation of $\vec{Q}$ within an element by a linear function and thus has nominally the same order of accuracy as the DG-p1 scheme. The main difference between the two schemes arises from the manner in which the gradient terms $\left(\vec{Q}_x, \vec{Q}_y, \vec{Q}_z\right)$ are computed. While a TVD scheme approximates the gradient terms using numerical differencing, a DG-p1 evaluates them by solving a set of PDEs derived using eqn.(2.3).

While TVD schemes are limited to 2nd order accuracy, DG schemes of arbitrary order of accuracy have been formulated. Third and higher-order ENO (Essentially non-Oscillatory) schemes which satisfy the TVD property almost everywhere have enjoyed only very limited success. On the other hand, in CEM solutions have been obtained for some three dimensional (3D) configurations using DG schemes of order up to eleven and it has been shown that it is possible to substantially increase accuracy and decrease turn-around time using higher order DG schemes.

3.0 DG Formulation for 1D Conservation Laws using Orthogonal Polynomials

In the initial stages of the contract, formulation for 1D conservation laws using both monomial and orthogonal basis functions were studied. The orthogonal basis functions are preferred over the monomials since they present two major advantages over the monomials, namely

1. The equations for the coefficients are uncoupled.
2. The formulation is hierarchical in the sense that to increase the order of the formulation from *n* to *n+1*, one needs to only add the equation for the *n*th coefficient. This property follows from the fact that the coefficients are uncoupled. On the other hand, in the case of monomials addition of a basis function modifies the governing equations for all the coefficients necessitating the reformulation of the scheme.

Another difference between the use of monomials and orthogonal polynomials arises from the fact that while in the case of monomials $\vec{Q}_e^0$ corresponds to $\vec{Q}$ at the element center, in the case of orthogonal polynomials it corresponds to the element average. In other words, in the case of monomials $\vec{Q}$ is expanded about the element center while in the case of orthogonal polynomials the expansion is about the element average.

The orthogonal polynomial basis functions, $v^i(\xi)$, employed in the formulation are the Legendre polynomials. They are obtained sequentially starting with the zeroth order polynomial by satisfying the following conditions:

$$\int_{-1}^{1} \nu^i \nu^j d\xi = \frac{2\delta_{ij}}{2i+1} \tag{3.1}$$

$$\nu^i(1) = 1 \tag{3.2}$$

$$\nu^i(\xi) = (-1)^i \nu^i(-\xi) \tag{3.3}$$

Eqns. (3.1), (3.2), and (3.3) lead to

$$v^0(\xi) = 1,\ v^1(\xi) = \xi,\ v^2(\xi) = \frac{3\xi^2 - 1}{2},\ v^3(\xi) = \frac{5\xi^3 - 3\xi}{2}, \text{...., and so on} \tag{3.4}$$

Following the presentation in section 2, we first expand the dependent variable vector, $\vec{Q}$, within an element as

$$\vec{Q}(x) = \sum_{i=0}^{n} \vec{Q}_e^i \cdot v_e^i(\xi) \tag{3.6}$$

where

$$\xi = 2 \cdot \frac{x - x_e}{h} \tag{3.7}$$

where x_e is the center of the element and h is its length. The dependent variables for the numerical scheme now consist of the (*n+1*) vectors $\vec{Q}_e^i, i = 0, n$. The weak form of the governing equations, eqn.(2.3), may now be written as

$$\frac{h}{2i+1} \frac{\partial \vec{Q}_e^i}{\partial t} + \vec{F}(1) - (-1)^i \vec{F}(-1) = \int_{-1}^{1} \frac{d\nu_e^i(\xi)}{d\xi} \cdot \vec{F}(\vec{Q}) \cdot d\xi + \int_{-1}^{1} v_e^i(\xi) \cdot \vec{S}(\vec{Q}, \xi, t) \cdot d\xi \tag{3.8}$$

Equation (2.6) is employed in the computation of $\vec{F}(1)$ and $F(-1)$ that involve computation of flux vector at an element boundary.

Orthogonality of the basis functions, $v_e^i(\xi)$, results in decoupling of the $\vec{Q}_e^i$'s on the left-hand side of eqn. (3.8). Orthogonality also simplifies higher-order implementations in the sense that the order of accuracy may be increased from *n* to *n+1* by just adding the equation for $\vec{Q}_e^{n+1}$. The equations for $\vec{Q}i$, *i=0,n,* remain unchanged. The integrals on the right-hand side (RHS) of eqn. (3.*8*) may be evaluated either using a Gaussian quadrature or by expanding the flux-vector $\vec{F}$ and the source-vector $\vec{S}$ about $\vec{Q}_e^0$. The Taylor series expansion for $\vec{F}$ about $\vec{Q}_e^0$ may be written as

$$f_i(\vec{Q}) = f_i(\vec{Q}_e^0) + \sum_j \frac{\partial f_i}{\partial q_j}\Delta q_j + \sum_j\sum_k \frac{\partial^2 f_i}{\partial q_j \partial q_k}\frac{\Delta q_j \cdot \Delta q_k}{2!} + \sum_j\sum_k\sum_l \frac{\partial^3 f_i}{\partial q_j \partial q_k \partial q_l}\frac{\Delta q_j \cdot \Delta q_k \cdot \Delta q_l}{3!} + \ldots\ldots\ldots\ldots \tag{3.9}$$

where f_i and qi refer to the i^{th} component of the flux vector and dependent variable vector respectively, and

$$\Delta q_i = q_i - q_{e\,i}^0 = \sum_{j=1}^{n} q_{e\,i}^j \cdot v_e^j(\xi) \tag{3.10}$$

$q_{e\,i}^j$ being the i^{th} component of $\vec{Q}_e^j$.

Using eqns. (3.9) and (3.10) the flux vector $\vec{F}$ may be expanded in terms of the basis functions as

$$\vec{F}(\xi) = \sum_{i=0}^{n} \vec{F}i\left(\vec{Q}_e^0, \vec{Q}_e^1, \ldots\ldots, \vec{Q}_e^1\right) \cdot v_e^i(\xi) \tag{3.11}$$

Substitution of eqn. (3.11) in eqn. (3.8) reduces the evaluation of the flux-vector integral on the RHS of eqn. (3.8) to the evaluation of the integrals

$$\int_{-1}^{1} \frac{\partial v_e^i}{\partial \xi} \cdot v_e^j(\xi) \cdot d\xi \quad i=0,n,\ j=0,n \tag{3.12}$$

The source-vector integral may also be obtained using the same procedure.

The final form of the DG equations up to p3 are the following:

$$\text{i=0:}\ h\vec{Q}0_t + \vec{F}(1) - \vec{F}(-1) = 2 \cdot \vec{S}0 \tag{3.13}$$

$$\text{i=1:}\ \frac{h}{3}\vec{Q}1_t + \vec{F}(1) + \vec{F}(-1) = 2 \cdot \vec{F}0 + \frac{1}{3} \cdot \vec{S}1 \tag{3.14}$$

$$\text{i=2:}\ \frac{h}{5}\vec{Q}2_t + \vec{F}(1) - \vec{F}(-1) = 2 \cdot \vec{F}1 + \frac{1}{5} \cdot \vec{S}2 \tag{3.15}$$

$$\text{i=3:}\ \frac{h}{7}\vec{Q}3_t + \vec{F}(1) + \vec{F}(-1) = 2 \cdot \left(\vec{F}2 - \vec{F}0\right) + \frac{1}{7} \cdot \vec{S}3 \tag{3.16}$$

where the flux is represented as

$$\vec{F} = \vec{F}0 \cdot v_e^0 + \vec{F}1 \cdot v_e^1 + \vec{F}2 \cdot v_e^2 + \vec{F}3 \cdot v_e^3 \tag{3.17}$$

and the source term is represented as

$$\vec{S} = \vec{S}0 \cdot \nu_e^0 + \vec{S}1 \cdot \nu_e^1 + \vec{S}2 \cdot \nu_e^2 + \vec{S}3 \cdot \nu_e^3 \tag{3.18}$$

Eqn. (3.8) can be rewritten generally as

$$\frac{\partial \vec{Q}}{\partial t} = RHS\left(\vec{Q}\right) \tag{3.19}$$

An explicit time update of eqn. (3.19) is done using the 4th order Runge-Kutta method. The following equations show how the time update is done.

$$\vec{Q}^{*}_{n+\frac{1}{2}} = \vec{Q}_n + \frac{\Delta t}{2} \cdot RHS\left(\vec{Q}_n\right) \tag{3.20}$$

$$\vec{Q}^{**}_{n+\frac{1}{2}} = \vec{Q}_n + \frac{\Delta t}{2} \cdot RHS\left(\vec{Q}^{*}_{n+\frac{1}{2}}\right) \tag{3.21}$$

$$\vec{Q}^{***}_{n+1} = \vec{Q}_n + \Delta t \cdot RHS\left(\vec{Q}^{**}_{n+\frac{1}{2}}\right) \tag{3.22}$$

$$\vec{Q}_{n+1} = \vec{Q}_n + \frac{\Delta t}{6} \cdot \left(RHS\left(\vec{Q}_n\right) + 2 \cdot RHS\left(\vec{Q}^{*}_{n+\frac{1}{2}}\right) + 2 \cdot RHS\left(\vec{Q}^{**}_{n+\frac{1}{2}}\right) + RHS\left(\vec{Q}^{***}_{n+1}\right) \right) \tag{3.24}$$

4.0 1-D and Quasi 1-D test cases

Formulations for 1D Euler, quasi-1D Euler, and 1D magneto-gas dynamics (MGD) equations are presented in this section.

4.1 Perfect Gas Euler Equations in 1-D

The partial differential equations (PDE) that result from the conservation principle for mass, momentum, and energy are

$$\vec{Q}_t + \vec{F}_x = 0 \qquad \vec{Q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix} \qquad \vec{F} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} \rho u \\ p + \rho u^2 \\ (e+p)u \end{bmatrix} = \begin{bmatrix} q_2 \\ p + \dfrac{q_2^2}{q_1} \\ (q_3 + p)\dfrac{q_2}{q_1} \end{bmatrix} \tag{4.1}$$

where $$p = (\gamma - 1)\left(e - \frac{1}{2}\rho u^2\right) = (\gamma - 1)\left(q_3 - \frac{q_2^2}{2q_1}\right)$$

where ρ is density, u, the velocity, p, the pressure, e, the total energy per unit volume, $\vec{Q}$, the vector of dependent variables (conserved quantities), and $\vec{F}$, the flux vector.

4.2 Quasi 1-D Perfect Gas Flow

The governing equations for an inviscid quasi-1D flow are given by

$$\vec{Q}_t + \vec{F}_x = \vec{S}$$

$$Q = \begin{bmatrix} \tilde{\rho} A \\ \tilde{\rho}\tilde{u} A \\ \tilde{e} A \end{bmatrix} \qquad F = \begin{bmatrix} \tilde{\rho} u A \\ \left(\tilde{p} + \tilde{\rho}\tilde{u}^2\right) A \\ \left(\tilde{e} + \tilde{p}\right)\tilde{u} A \end{bmatrix} \qquad \vec{S} = \begin{bmatrix} 0 \\ \tilde{p}\dfrac{dA}{dx} \\ 0 \end{bmatrix}$$

Let $\rho = \tilde{\rho} A,\ \ u = \tilde{u},\ \ e = \tilde{e} A,\ \ p = \tilde{p} A,$ then (4.2)

$$Q = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix} \qquad \tilde{F} = \begin{bmatrix} \rho u \\ \left(p + \rho u^2\right) \\ \left(e + p\right) u \end{bmatrix} \qquad \vec{S} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 0 \\ p \cdot g \\ 0 \end{bmatrix}$$

$$\textit{where} \quad g = \frac{1}{A}\frac{dA}{dx}$$

where A is the area of the nozzle. That is, the governing equations for a quasi-1D flow (eqn. 4.2) are same as the 1D Euler equations (eqn. 4.1) except for the fact that the momentum equation includes a source term and the fact that ρ, e, and p are scaled by the area A.

The presence of the source term $\vec{S}$ necessitates evaluation of the following integrals in the weak form of the governing equations:

$$\int_{-\frac{h}{2}}^{\frac{h}{2}} v_e^i \cdot s_2 \cdot dx\,,\ \mathrm{i = 0, n - 1} \tag{4.3}$$

This integral may be transformed to

$$\frac{h}{2}\int_{-1}^{1} v_e^i \cdot s_2 \cdot d\xi\,,\ \mathrm{i = 0, n - 1} \tag{4.4}$$

where the source term s_2 is expanded in terms of the basis functions as

$$s_2(\xi) = \sum_{j=0}^{n} s_2^j\left(\vec{Q}_e^0, \vec{Q}_e^0, \ldots\ldots, \vec{Q}_e^n\right) \cdot v_e^j(\xi) \tag{4.5}$$

This reduces the evaluation of the source term integral on the RHS of eqn. (4.2) to the evaluation of the integrals

$$\int_{-1}^{1} \mathrm{v}_e^i(\xi) \cdot v_e^j(\xi) \cdot d\xi \quad i=0,n,\ j=0,n \tag{4.6}$$

The final form of the source term in eqn. (3.18) is

$$\vec{S}j = \begin{pmatrix} 0 \\ s_2^j \\ 0 \end{pmatrix} \tag{4.7}$$

4.3 Governing Equations for MGD Flows in 1-D

Simulation of inviscid flow of a perfect-gas in the presence of a magnetic field (inviscid MGD flow) involves 8 dependent variables, namely, density ρ, velocity components (u,v,w), energy per unit mass, e, and three components of the magnetic field, (B_x, B_y, B_z). In the case of one-dimensional flows, the derivatives in two coordinate directions, for examples, the y and z

directions, vanish identically. When there is no magnetic field present, this implies that the v and w components of velocity must remain constant in the x-direction. On the other hand, when a magnetic field is applied to the flow of a conducting fluid, Lorentz forces are produced, that can cause an x-variation of v and w components. The only dependent variable that vanishes identically is the *x* component of the magnetic field, B_x. (The equation set for ideal MHD is over-determined, and requires that the magnetic field is divergence free. In 1-D this implies that $\boldsymbol{B_x} \equiv \text{constant}$, since $\frac{\partial B_y}{\partial y} \equiv \frac{\partial B_z}{\partial z} \equiv 0 \Rightarrow \frac{\partial B_x}{\partial x} \equiv 0$.)

Further details on the conservation laws in MGD may be found in any standard text on the subject, e.g., Sutton and Sherman [5]. The governing equations for inviscid, 1D, MGD flows may be written in conservation form as

$$Q_t + F_x = 0 \tag{4.7}$$

where the vector of dependent variables, *Q*, and the flux vector, *F*, are given by

$$\vec{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ B_y \\ B_z \\ e \end{bmatrix} \qquad \vec{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p^* \\ \rho uv - B_x B_y \\ \rho uw - B_x B_z \\ B_y u - B_x v \\ B_z u - B_z w \\ \left(e + p^*\right)u - B_x\left(B_x u + B_y v + B_z w\right) \end{bmatrix} \tag{4.8}$$

Total energy, *e*, and modified pressure, p^* are defined by

$$e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right) + \frac{1}{2}\left(B_x{}^2 + B_y{}^2 + B_z{}^2\right) \tag{4.9}$$

$$p^* = p + \frac{1}{2}\left(B_x{}^2 + B_y{}^2 + B_z{}^2\right) \tag{4.10}$$

Eigen values of the 1-D inviscid flux may be computed from the above relations, and are required to estimate numerical dissipation in the Lax-Friedrichs method. They are:

$$\lambda = \begin{Bmatrix} u \pm c_f \\ u \pm c_a \\ u \pm c_s \\ u \end{Bmatrix}, \text{ with } \quad \begin{aligned} c_a &= \sqrt{b_x} \\ c_{f,s} &= \sqrt{\frac{1}{2}\left(a^{*2} \pm \sqrt{a^{*4} - 4a^2 b_x{}^2}\right)} \end{aligned} \tag{4.11}$$

$$b_x = \frac{B_x}{\sqrt{\rho}} \quad b_y = \frac{B_y}{\sqrt{\rho}} \quad b_z = \frac{B_z}{\sqrt{\rho}} \quad b = \sqrt{{b_x}^2 + {b_y}^2 + {b_z}^2}$$

$$a = \sqrt{\frac{\gamma P}{\rho}} \quad a^* = \sqrt{a^2 + b^2}$$

4.4 Limiters

In the numerical solution of partial differential equations that govern the physics of fluid motion limiters are a necessary evil. While they do reduce the accuracy of a numerical scheme, they are required to ensure stability when discontinuities and high curvature regions are encountered. Even though Atkins and Shu state in one of their papers [6] that they have been able to obtain stable solutions for a non-linear problem in 1D with shocks, their solutions for DG-p2 exhibit typical behavior of a solution obtained without limiters (overshoots and undershoots in the shock region) clearly confirming the need for limiters.

Limiters enter into a computational process that employs a piecewise continuous approximation for the dependent variables when integrals at element boundaries are evaluated. Evaluation of such an integral involves computation of the integrant at element boundaries by interpolating the piecewise continuous approximation. This process invariably leads to two different values for the integrant computed from the interpolation performed for the two neighboring elements. Such a discontinuity at an element boundary is usually resolved using a Riemann solver. Limiters are required to ensure that the interpolation process does not result in new maxima or minima. In our formulation, we employ a minmod limiter [7] for each of the coefficients $Q_e^i(t)$ in eqn. (2.2). More research is needed to optimize limiting to ensure stability of the scheme without compromising accuracy.

5.0 1-D Results

Results from the application of the 1D DG scheme for the simulation of

1. a simple scalar wave propagation,
2. inviscid propagating density wave,
3. inviscid flow in a shock tube,
4. inviscid, quasi-1D flow in a convergent-divergent nozzle,
5. MGD flow with an intermediate shock, and
6. Brio-Wu problem

are presented in the following sections. Note that the order in which the results are presented represent the increasing order of difficulty involved in simulating a simple scalar wave to an MGD flow that involves discontinuities.

The first example considered here, namely *propagation of a simple wave*, is probably the simplest form of conservation law that one may encounter. In this case the vector of dependent variable has just one element. In other words, the governing equation involves a scalar conservation law. The flux vector is just a linear function of the dependent variable vector. The solution involves a simple wave propagating at a constant speed.

The second example involves a system of conservation laws. The equations solved are the one-dimensional Euler equations (eqn. 4.1). This case also involves a simple traveling wave. By setting $u \equiv 1$ and $p \equiv 1$, the solution is forced to correspond to a density wave traveling at a constant speed. This example serves the purpose of validating the implementation of a 1D, DG solver for Euler equations.

The flow in a shock tube is a classical problem employed in computational fluid dynamics (CFD) to evaluate the accuracy of a numerical scheme in simulating a system of equations that involve multiple waves traveling at non-constant speeds.

Inviscid, quasi-1D flow adds the complexity of a source term to the governing equations and serves the purpose of evaluating the accuracy of a numerical scheme in dealing with source terms.

Intermediate MHD shocks are discontinuities across which the tangential magnetic field component changes sign and the shock frame velocity of the flow changes from super to sub-Alfvenic across the shock. We select this problem for study, since it stands to benefit from higher order simulations and possesses smooth solutions (when used with periodic BCs in 1-D) until the formation of the shock.

The Brio and Wu [2] 1-D MHD shock tube validation case represents an important porting of shock tube validation studies (such as that of Sod, Van Leer, etc. in perfect gases,) to MHD.

5.1 Scalar Wave Propagation in 1D

This simple time-dependent problem demonstrates the hierarchical improvement in simulation accuracy attainable using the DG-p0, DG-p1, and DG-p2 schemes. In this case the governing equation is given by

$$u_t + u_x = 0, \quad -0.1 \le x \le 2.0 \tag{5.1}$$

with a cubic polynomial as the initial condition. That is,

$$u(x,0) = \begin{bmatrix} 0, & -0.1 \le x \le -0.05 \\ -16000 \cdot x^3 + 1200 \cdot x^2 + 1 & -0.05 \le x \le 0 \\ -16000 \cdot x^3 + 1200 \cdot x^2 + 1 & 0 \le x \le 0.05 \\ 0 & 0.05 \le \mathrm{x} \le 2.0 \end{bmatrix} \tag{5.2}$$

The boundary conditions employed are

$$u(-0.1,t) = u_x(-0.1,t) = u(2,t) = u_x(2,t) = 0 \tag{5.2}$$

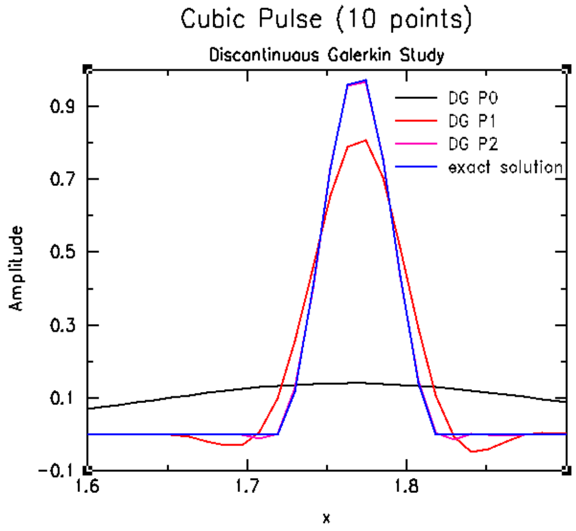


Figure 1. Solution for t=1.6

The grid employed has a resolution of $dx = 0.01$. Solutions obtained for $t = 1.6$ using DG-p0, DG-p1, and DG-p2 are compared with the exact solution in fig.1. The exact solution is nothing but a simple wave traveling to the right at a constant speed. As such the solution at $t = 1.6$ is the same as the solution at $t = 0$ shifted to the right by 1.6.

Hierarchical nature of the simulation accuracy may be seen from the fact that while the DG-p2 solution agrees with the exact solution almost exactly, DG-p1 solution is visibly different from the exact solution and DG-p0 solution has an unacceptably large error. The dissipative and disspersive nature of the errors may be observed from the decrease in amplitude and increase in wave length that accompany a decrease in the order of the scheme.

5.2 Propagating Density Wave

In the last example, we demonstrated the accuracy of a DG scheme in simulating a scalar traveling wave. The governing equation involved just one PDE (partial differential equation). Here we consider a system of PDEs representing three conservation laws, namely, the 1D, Euler equations (eqn. 4.1). In order to simplify the problem and employ this case as a test case for the verification of our implementation of a DG scheme for 1D, Euler equations, we force u and p to be identically equal to 1 so that the solution involves a simple density wave propagating at a constant speed. In reality this case is no different from the first example as far as the solution is concerned. The only difference is that we use a code developed for solving the 1D, Euler equations.

The 1D, Euler equations are solved in the domain $-1 \le x \le 1$, with the initial condition for density given by

$$\rho(x,0) = 2 + \sin(\pi x)$$

With periodic boundary conditions, $u \equiv 1$, and $p \equiv 1$ this problem has the following exact solution:

$$\rho = 2 + \sin(\pi(x - ut))$$

The solution for this case at t=100 is shown in fig. 2. It may be seen from this figure that the p2 scheme preserves the wave better than the p1 scheme indicating the improved accuracy that is possible with the p2 scheme.

Error analysis performed for three different grid spacings is shown in Table 1. It may be seen from the table that the error for the p2 scheme is 2 to 3 orders of magnitude smaller. The order of convergence is also higher for the p2 scheme.

No. of grid points	P1: error	order	P2: error	order
32	2.5e-1		2.0e-3	
64	3.5e-2	2.8	1.1e-4	4.1
128	4.5e-3	2.9	9.6e-6	3.6

Table 1: Error for the p1 and p2 schemes for different grid resolutions

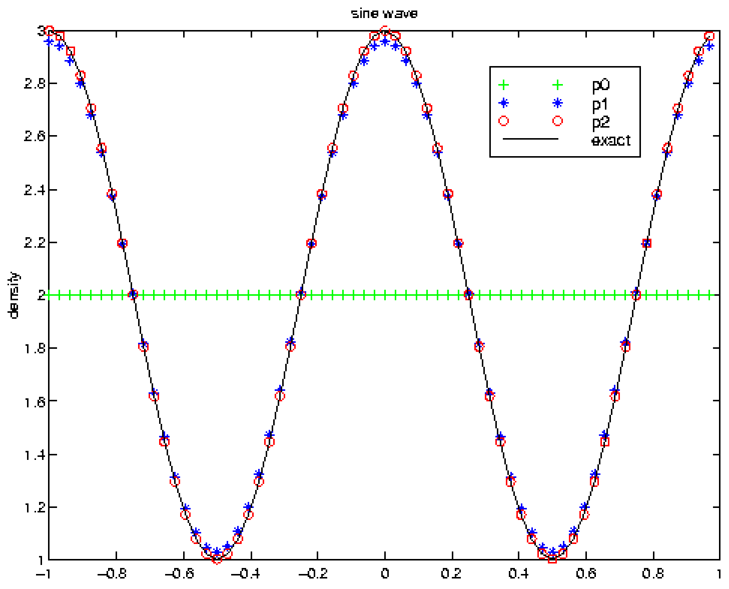


Figure 2. Solution for a simple density wave at t=100.

One of the major benefits that one may expect to derive from application of a higher order scheme to an unsteady problem involves predicting solutions for large *t*. In order to assess the improvement in accuracy attainable with a fourth order scheme (DG-p3), the density wave was propagated to *t=50,000*. The results obtained using the p2 and p3 schemes are compared in fig.3.

The grid resolution employed corresponds to 20 points per wave. It may be seen from this figure that the p3 scheme performs much better than p2 scheme justifying the use of higher-order accuracy in similar problems.

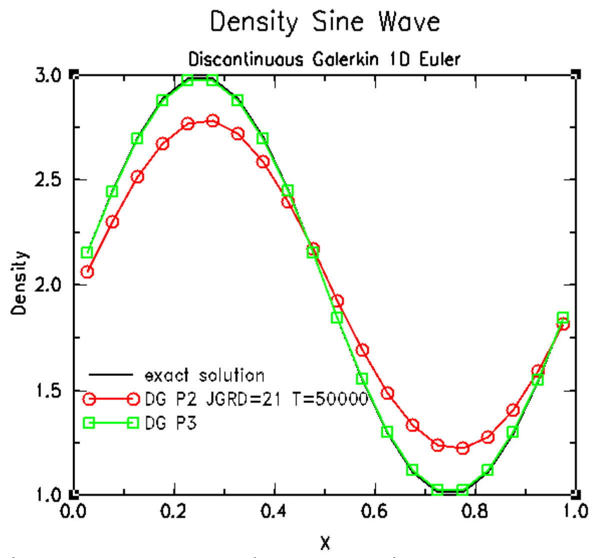


Figure 3. DG-p2 and DG-p3 schemes compared.

5.3 Shock Tube

The flow in this example starts with a stationary gas in a tube partitioned by a diaphragm. The gas on the right-hand side of the diaphragm is at a higher pressure $(p4)$ and density $(\rho 4)$ than the gas on the left-hand side $(p1 \text{ and } \rho 1)$. Simulations were carried out with pressure and density ratios given by

$$\frac{p4}{p1} = 2\,, \quad \frac{\rho 4}{\rho 1} = 2$$

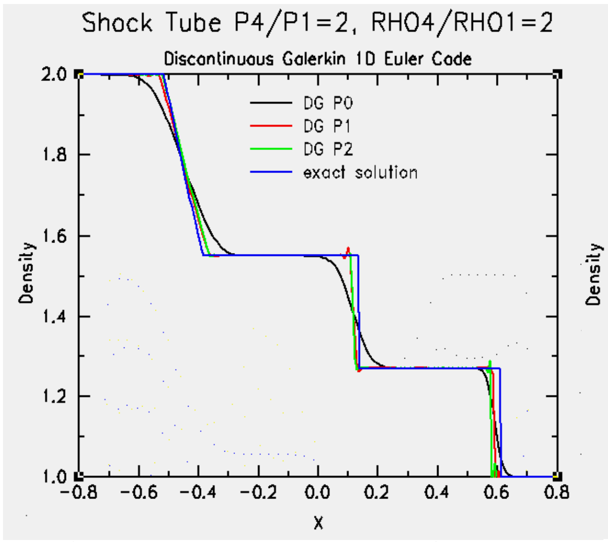
When the diaphragm is punctured at $t = 0$, a shock wave propagates into the high pressure region while an expansion wave and a contact discontinuity propagate into the low pressure region. The simulation discussed here employed a computational domain given by: $-0.5 \leq x \leq 0.5$. The diaphragm was located at $x = 0$. The grid employed had 300 elements.

Solutions obtained using DG-p0, DG-p1, and DG-p2 are compared with the exact solution in fig.4. While the figure shows a substantial improvement in accuracy as the order of the scheme is increased from 1 (DG-p0) to 2 (DG-p1), the solution for 3rd order (DG-p2) scheme appears to be no better than the solution for the 2nd order scheme. This may be due to the fact that unlike the last two examples the 2nd derivative in this case is identically zero for a substantial portion of the computational domain decreasing the role of the parabolic term in the basis function.

Fig. 5 shows the effect of the CFL number on the solution for the DG-p1 scheme. While CFL=0.1 and CFL=0.3 produce almost identical solutions, CFL=0.4 results in an oscillatory behavior indicating that the simulation is unstable. This behavior is consistent with the fact that the CFL limit for a DG-pn scheme is given by [8]

$$CFL \le \frac{1}{2 \cdot n + 1} \tag{37}$$

and as such the DG-p1 scheme is unstable for $\mathrm{CFL} \ge \frac{1}{3}$.



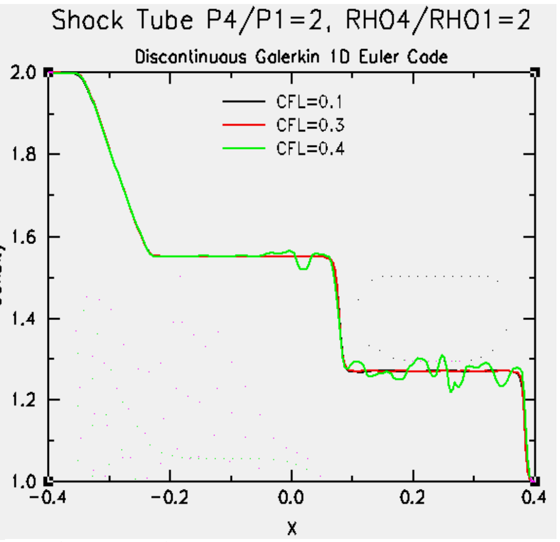


Figure 4. DG-p0, DG-p1, and Dg-p2 compared Figure 5. DG-p1; effect of CFL

5.4 Quasi-1D Flow

So far we have considered a simple wave resulting from a scalar conservation law, a density wave that represents the solution to a system of conservation laws, and a flow that involves multiple waves traveling at non-constant speeds. In this section we present solutions for inviscid flow in a quasi-1D convergent-divergent nozzle which involves solving a system of conservation laws with an added complexity of a source term.

The convergent-divergent nozzle selected for this example has cross-sectional area represented by a cubic polynomial. The computational domain extends from $x = -1$ to $x = 1$. The area distribution is symmetric about $x = 0$. Area is minimum at $x = 0$. The area ratio, that is the ratio between the area at $x = 0$ and area at $x = -1$ (or $x = 1$) is equal to 0.9. The nozzle contour was defined in such a way that the slope was zero at the entrance ($x = -1$), exit ($x = 1$), and throat ($x = 0$) of the nozzle. Flow conditions were chosen to ensure that the flow did not choke, that is the flow remained either supersonic or subsonic everywhere depending on the flow at nozzle entrance.

The results obtained from our study are shown in fig.6. The DG-p0 scheme has a slope of 1 while DG-p1 has a slope of 1.92 which corresponds to orders of accuracy of 1 and 1.92 respectively. We have not yet been able to obtain satisfactory performance with the DG-p2 scheme. The error for the DG-p2 scheme (not shown in the figure) is slightly less that for DG-p1 but only slightly. We believe that this is due to the fact that the error for the DG-p1 solution is already very near machine zero for.

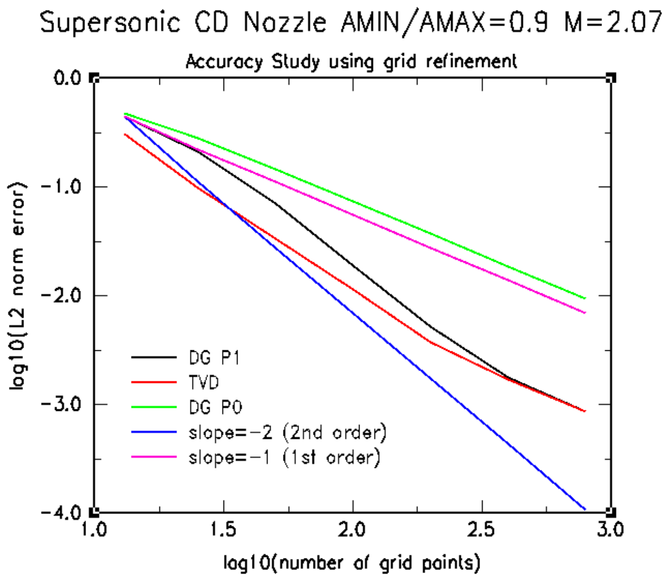


Figure 6. Error Analysis

5.5 Intermediate Shock Simulation

Intermediate MHD shocks are discontinuities across which the tangential magnetic field component changes sign and the shock frame velocity of the flow changes from super to sub-Alfvenic across the shock. Wu [3] first demonstrated the feasibility of formation of intermediate shocks in a MHD flow through nonlinear wave steepening from continuous waves. We select this problem for study, since it stands to benefit from higher order simulations and possesses smooth solutions (when used with periodic BCs in 1-D) until the formation of the shock. Wu [3] employed the following simple wave solution of the MHD equations as the initial conditions:

$$B_y = 0.5 \cdot \sin(2\pi x)$$

$$\frac{\partial \rho}{\partial B_y} = \frac{B_y}{\left(c_s - a^2\right)},$$

$$\frac{\partial(\rho u)}{\partial B_y} = \frac{(u + c_s) \cdot B_y}{\left(c_s{}^2 - a^2\right)}, \quad \frac{\partial(\rho v)}{\partial B_y} = \frac{\left(a^2 B_x - B_x c_s{}^2 + c_s v B_y\right)}{c_s \cdot \left(c_s{}^2 - a^2\right)}$$

$$\frac{\partial p}{\partial B_y} = \frac{a^2 \cdot B_y}{C_s^2 - a^2}$$

$$w = 0 \quad B_x = 1 \quad B_z = 0$$

Jiang and Wu [9] obtained numerical solutions for this intermediate shock problem using a 3rd order ENO and a 5th order WENO schemes. Their results showed that the 3rd order ENO and the 5th order WENO schemes performed quite well when marched in time to obtain solution for t=0.25 (fig. 7). At this particular instance in time the y-component of the magnetic field, B_y, is quite smooth and still far from steepening. On the other hand, at t=0.5 (fig. 8), B_y profile exhibits large gradients showing a tendency to form a shock. Their error analysis, when applied to the whole computational domain showed poor results. When they restricted their error analysis to smooth regions of the flow, they were able to show that their algorithm had the desired order of accuracy.

Solutions obtained from our numerical simulation for this problem are shown in figs. 7-8. Solutions have been obtained using various grids with spatial resolution as coarse as 0.1 (10 elements) to as fine as 0.00039 (2560 elements). The y-component of the magnetic field at t=0.25, shown in fig. 7, appears to have been quite well predicted by all grids except the coarsest. The plot of the L1-norm of the error, computed using the finest grid solution as the exact solution, indicates that the 2nd (p1) and 3rd order (p2) schemes have the same level of accuracy and match the accuracy of Jiang and Wu's 3rd order ENO scheme. The error analysis for t=0.5 indicates that the 3rd order (p2) DG algorithm is more accurate than the 2nd order scheme. More interestingly, both the 2nd and 3rd order DG algorithms perform better than even the 5th order WENO scheme [9].

The fact that the p2 solution is more accurate than p1 solution for t=0.5 and the fact that for t=0.25 the two solutions have almost the same order of accuracy is probably due to the fact that the contribution from $v_e^2(\xi)$ is not significant when the solution is smooth and becomes significant as the B_y profile starts steepening.

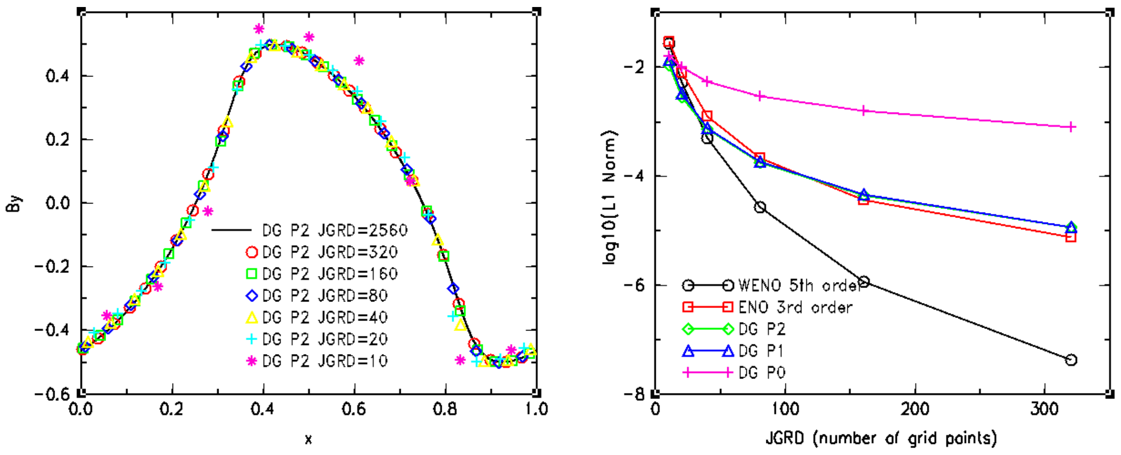


Figure 7. Solution for t=0.25

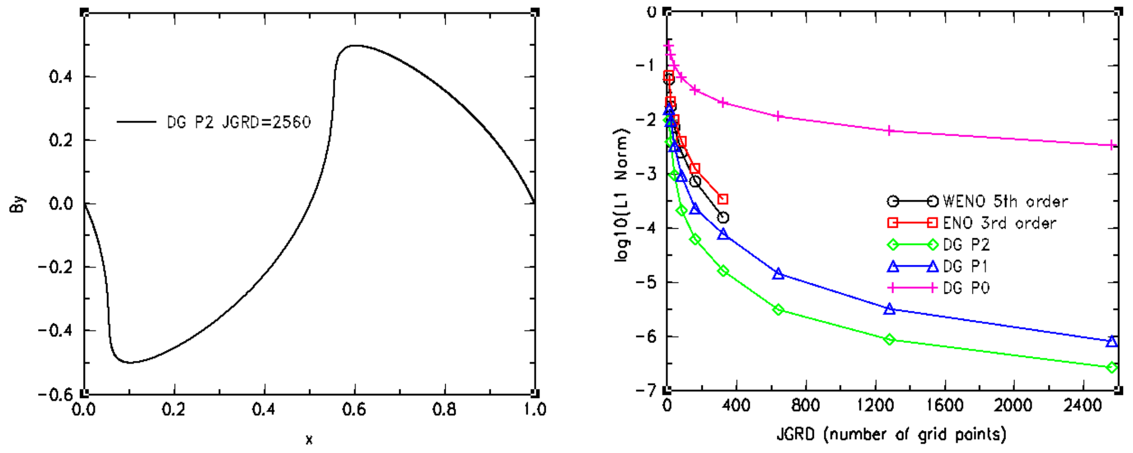


Figure 8. Solution for t=0.5

5.6 The Brio-Wu Shock-Tube Problem

Brio and Wu [2] presented the first implementation of Roe's Riemann solver in 1-D MHD problems and obtained a Roe linearization for the case of $\gamma = 2$. The shock tube problem used by them to demonstrate their solver represents an important porting of shock tube validation studies (such as that of Sod, Van Leer, etc. in perfect gases,) to MHD.

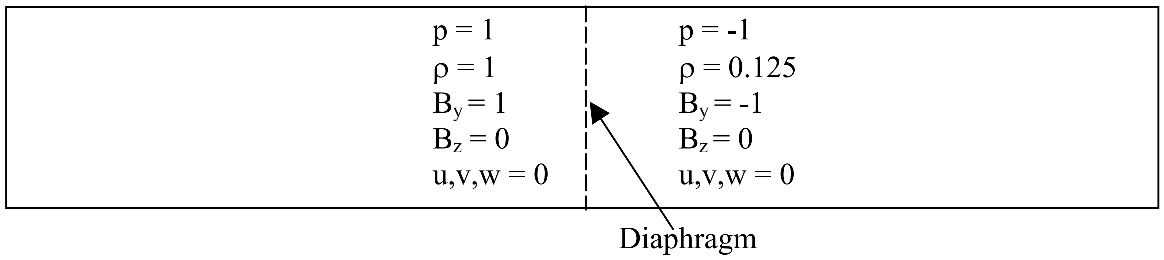


Figure 9. Initial Conditions for the shock-tube problem

The initial conditions employed in this problem are shown in fig. 9. Distributions of pressure, density, and y-component of the magnetic field obtained from our numerical simulation using a fine grid (800 elements) and a coarse grid (100 elements) are shown in figs. 10-11. Our fine grid solutions using 2nd order (p1) and 3rd order (p2) schemes were compared with those from a Kinetic Flux Vector Splitting scheme developed by Xu [10] and implemented earlier in MHD related work in HyPerComp (Munipalli and Shankar [11]). These two results match closely. Coarse-grid solutions are compared with the 3rd order fine-grid solution in Fig. 11. It may be seen from this figure that the 2nd and 3rd order solutions are almost identical and show a marked improvement over the 1st order solution. As in the case of intermediate shock solution for t=0.25, the 2nd derivatives are almost non existent in many regions of the flow contributing to the fact that the 2nd and 3rd order solutions are almost identical.

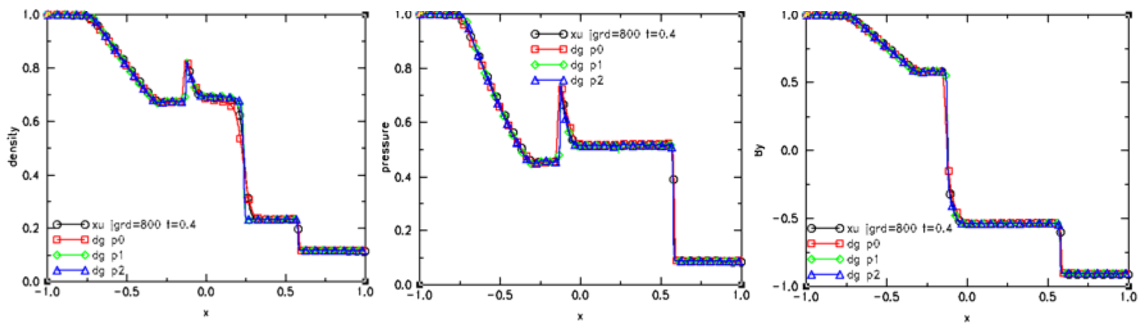


Figure 10. Fine-grid (800 grid elements) solution for t=0.4.

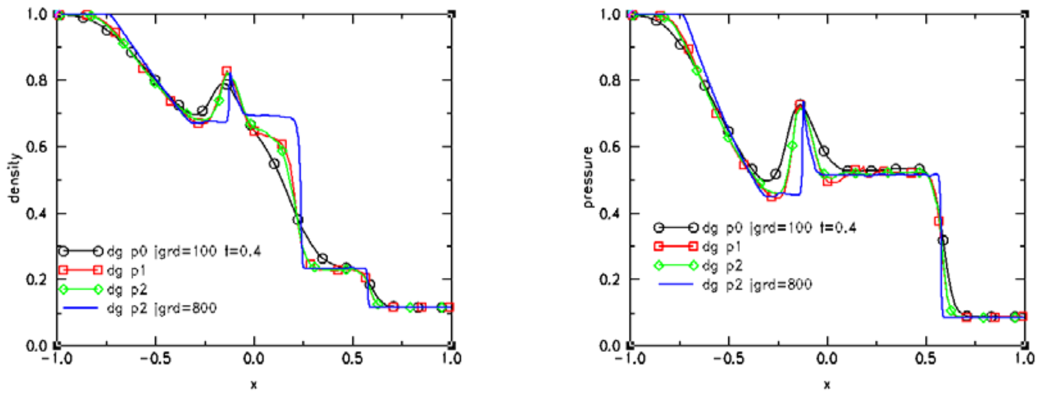


Figure 11. Coarse-grid (100 grid elements) solution for t=0.4.

6.0 Point Implicit Scheme with Orthogonal Polynomials

All the computations discussed so far were carried out using a 4th order Runge-Kutta explicit time marching scheme. The need for an implicit scheme arises from the fact that the CFL limit for an explicit DG-pn scheme is given by [8]

$$CFL \leq \frac{1}{2n+1} \tag{6.1}$$

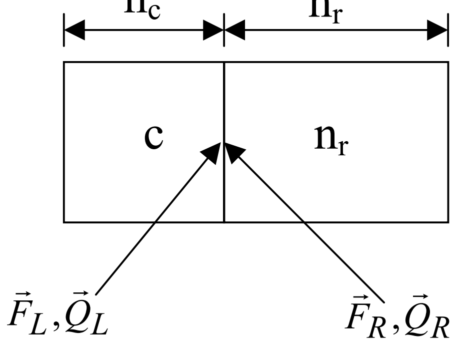
This imposes a very stringent limitation on the allowable time-step. Implicit schemes, on the other hand, do not have any CFL limit and the allowable time step is determined only by the time accuracy required by the problem under consideration. For steady state simulations, ability to employ large values for the time-step will result in fast turn-around time. Also, when a non-uniform grid is employed, allowable time-step is determined by the finest grid element which could be orders of magnitude smaller than the coarsest, especially for viscous flows. Hence, it is highly desirable to develop an implicit scheme.

In order to advance the solution from time level *n* to level (*n+1*) using an implicit scheme, all the terms in the discretized governing equations (eqn. 3.8) must be evaluated at time level (*n+1*). Specifically, all the flux terms and the source terms need to be computed at time level (*n+1*).

Expressions for $\vec{F}^{n+1}(1) = \vec{\mathrm{F}}^{\mathrm{n+1}}\left(\frac{\mathrm{h_c}}{2}\right)$ and $\vec{F}^{n+1}(-1) = \vec{\mathrm{F}}^{\mathrm{n+1}}\left(\frac{-\mathrm{h_c}}{2}\right)$ in eqn. 3.8 (flux at the right and left boundary respectively) using the scalar dissipation scheme (eqn. 2.6) and orthogonal polynomials as basis functions is presented in this section.

6.1 Derivation of the scheme

For the **right face**,



$c \Rightarrow$ cell, $n_r \Rightarrow$ right neighbor

$$\vec{Q}_L^n = \vec{Q}0c^n + \vec{Q}1c^n$$

$$\vec{Q}_L^{n+1} = \vec{Q}_L^n + \left(\frac{\partial \vec{Q}0c}{\partial t} + \frac{\partial \vec{Q}1c}{\partial t}\right)\Delta t$$

$$\vec{Q}_R^n = \vec{Q}0{n_r}^n - \vec{Q}1{n_r}^n$$

$$\vec{Q}_R^{n+1} = \vec{Q}_R^n + \left(\frac{\partial \vec{Q}0n_r}{\partial t} - \frac{\partial \vec{Q}1n_r}{\partial t}\right)\Delta t$$

$$\vec{F}^{n+1}\left(\frac{h_c}{2}\right) = \frac{1}{2}\left(\vec{F}_L^{n+1} + \vec{F}_R^{n+1}\right) - \frac{1}{2}|\lambda|_{\max}\left(\vec{Q}_R^{n+1} - \vec{Q}_L^{n+1}\right)$$

$$\vec{F}_L^{n+1} = \vec{F}_L^n + \left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)^n_{\frac{h_c}{2}}\left(\frac{\partial \vec{Q}0c}{\partial t} + \frac{\partial \vec{Q}1c}{\partial t}\right)$$

$$\vec{F}_R^{n+1} = \vec{F}_R^n + \left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)^n_{-\frac{h_r}{2}}\left(\frac{\partial \vec{Q}0n_r}{\partial t} - \frac{\partial \vec{Q}1n_r}{\partial t}\right)$$

$$\vec{F}^{n+1}\left(\frac{h_c}{2}\right)=\frac{1}{2}\left(\vec{F}_L^n+\vec{F}_R^n\right)-\left|\lambda\right|_{\max}\left(\vec{Q}_R^n-\vec{Q}_L^n\right)$$

$$+\frac{1}{2}\left[\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)^n_{\frac{h_c}{2}}\left(\frac{\partial\vec{Q}0c}{\partial t}+\frac{\partial\vec{Q}1c}{\partial t}\right)+\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)^n_{-\frac{h_r}{2}}\left(\frac{\partial Q0n_r}{\partial t}-\frac{\partial Q1n_r}{\partial t}\right)\right]\Delta t$$

$$-\left|\lambda\right|_{\max}\left[\frac{\partial Q0n_r}{\partial t}-\frac{\partial Q1n_r}{\partial t}-\frac{\partial\vec{Q}0c}{\partial t}-\frac{\partial\vec{Q}1c}{\partial t}\right]$$

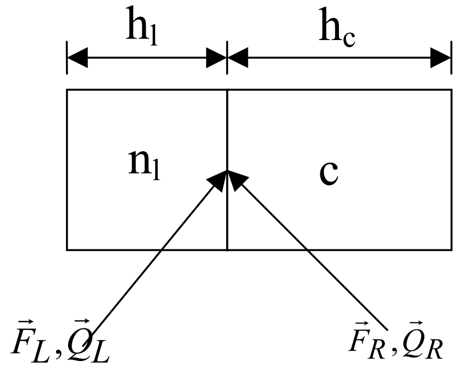
$$=\vec{F}^n\left(\frac{h_c}{2}\right)+\frac{1}{2}\left[\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)^n_{\frac{h_c}{2}}+\left|\lambda\right|_{\max}\right]\frac{\partial\vec{Q}0c}{\partial t}\Delta t+\frac{1}{2}\left[\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)^n_{-\frac{h_r}{2}}-\left|\lambda\right|_{\max}\right]\frac{\partial\vec{Q}0n_r}{\partial t}\Delta t$$

$$+\frac{1}{2}\left[\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)^n_{\frac{h_c}{2}}+\left|\lambda\right|_{\max}\right]\frac{\partial\vec{Q}1c}{\partial t}\Delta t-\frac{1}{2}\left[\left(\frac{\partial\vec{\mathrm{F}}}{\partial\vec{\mathrm{Q}}}\right)^n_{-\frac{h_r}{2}}-\left|\lambda\right|_{\max}\right]\frac{\partial\vec{Q}0n_r}{\partial t}\Delta t$$

$$\phi(x)=\frac{1}{2}\left[\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)+\left|\lambda\right|_{\max}\right]^n_x\qquad\sigma(x)=\frac{1}{2}\left[\left(\frac{\partial\vec{F}}{\partial\vec{Q}}\right)-\left|\lambda\right|_{\max}\right]^n_x$$

$$\vec{F}^{n+1}\left(\frac{h_c}{2}\right)=\vec{F}^n\left(\frac{h_c}{2}\right)+\Delta t\phi\left(\frac{h_c}{2}\right)\left(\frac{\partial\vec{Q}0c}{\partial t}+\frac{\partial\vec{Q}1c}{\partial t}\right)+\Delta t\sigma\left(-\frac{h_r}{2}\right)\left(\frac{\partial\vec{Q}0n_r}{\partial t}-\frac{\partial\vec{Q}1n_r}{\partial t}\right)\quad(6.2)$$

For the **left face**,



$$c\Rightarrow\text{cell},\quad n_l\Rightarrow\text{left neighbor}$$

$$\vec{Q}_L^n=\vec{Q}0{n_l}^n+\vec{Q}1{n_l}^n$$

$$\vec{Q}_L^{n+1}=\vec{Q}_L^n+\left(\frac{\partial\vec{Q}0n_l}{\partial t}+\frac{\partial\vec{Q}1n_l}{\partial t}\right)\Delta t$$

$$\vec{Q}_R^n = \vec{Q}0c^n - \vec{Q}1c^n$$

$$\vec{Q}_R^{n+1} = \vec{Q}_R^n + \left(\frac{\partial \vec{Q}0c}{\partial t} - \frac{\partial \vec{Q}1c}{\partial t}\right)\Delta t$$

$$\vec{F}^{n+1}\left(-\frac{h_c}{2}\right) = \frac{1}{2}\left(\vec{F}_L^{n+1} + \vec{F}_R^{n+1}\right) - \frac{1}{2}\left|\lambda\right|_{\max}\left(\vec{Q}_R^{n+1} - \vec{Q}_L^{n+1}\right)$$

$$\vec{F}_L^{n+1} = \vec{F}_L^n + \left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{\frac{h_l}{2}}^n \left(\frac{\partial \vec{Q}0n_l}{\partial t} + \frac{\partial \vec{Q}1n_l}{\partial t}\right)$$

$$\vec{F}_R^{n+1} = \vec{F}_R^n + \left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{-\frac{h_c}{2}}^n \left(\frac{\partial \vec{Q}0c}{\partial t} - \frac{\partial \vec{Q}1c}{\partial t}\right)$$

$$\begin{aligned}
\vec{F}^{n+1}\left(-\frac{h_c}{2}\right) &= \frac{1}{2}\left(\vec{F}_L^n + \vec{F}_R^n\right) - \left|\lambda\right|_{\max}\left(\vec{Q}_R^n - \vec{Q}_L^n\right) \\
&\quad + \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{\frac{h_l}{2}}^n \left(\frac{\partial \vec{Q}0n_l}{\partial t} + \frac{\partial \vec{Q}1n_l}{\partial t}\right) + \left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{-\frac{h_c}{2}}^n \left(\frac{\partial \vec{Q}0c}{\partial t} - \frac{\partial \vec{Q}1c}{\partial t}\right)\right]\Delta t \\
&\quad - \left|\lambda\right|_{\max}\left[\frac{\partial \vec{Q}0c}{\partial t} - \frac{\partial \vec{Q}1c}{\partial t} - \frac{\partial \vec{Q}0n_l}{\partial t} - \frac{\partial \vec{Q}1n_l}{\partial t}\right] \\
&= \vec{F}^n\left(-\frac{h_c}{2}\right) + \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{\frac{h_l}{2}}^n + \left|\lambda\right|_{\max}\right]\frac{\partial \vec{Q}0n_l}{\partial t}\Delta t + \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{-\frac{h_c}{2}}^n - \left|\lambda\right|_{\max}\right]\frac{\partial \vec{Q}0c}{\partial t}\Delta t \\
&\quad + \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{\frac{h_l}{2}}^n + \left|\lambda\right|_{\max}\right]\frac{\partial \vec{Q}1n_l}{\partial t}\Delta t - \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right)_{-\frac{h_c}{2}}^n - \left|\lambda\right|_{\max}\right]\frac{\partial \vec{Q}1c}{\partial t}\Delta t
\end{aligned}$$

$$\phi(x) = \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right) + \left|\lambda\right|_{\max}\right]_x^n \qquad \sigma(x) = \frac{1}{2}\left[\left(\frac{\partial \vec{F}}{\partial \vec{Q}}\right) - \left|\lambda\right|_{\max}\right]_x^n$$

$$\vec{F}^{n+1}\left(-\frac{h_c}{2}\right) = \vec{F}^n\left(-\frac{h_c}{2}\right) + \Delta t\phi\left(\frac{h_l}{2}\right)\left(\frac{\partial \vec{Q}0n_l}{\partial t} + \frac{\partial \vec{Q}1n_l}{\partial t}\right) + \Delta t\sigma\left(-\frac{h_c}{2}\right)\left(\frac{\partial \vec{Q}0c}{\partial t} - \frac{\partial \vec{Q}1c}{\partial t}\right) \tag{6.3}$$

PDE for $\vec{Q}0$:

$$h_c \cdot \frac{\partial \vec{Q}0c}{\partial t} + \vec{F}\left(\frac{h_c}{2}\right)^{n+1} - \vec{F}\left(-\frac{h_c}{2}\right)^{n+1} = 0 \tag{6.4}$$

From equations (6.2), (6.3), and (6.4), we get

$$\begin{aligned} & h_c \cdot \frac{\partial \vec{Q}0c}{\partial t} + \vec{F}\left(\frac{h_c}{2}\right)^{n} - \vec{F}\left(-\frac{h_c}{2}\right)^{n} \\ & \quad + \Delta t\left(\phi\left(\frac{h_c}{2}\right) - \sigma\left(-\frac{h_c}{2}\right)\right)\frac{\partial \vec{Q}0c}{\partial t} + \Delta t\left(\phi\left(\frac{h_c}{2}\right) + \sigma\left(-\frac{h_c}{2}\right)\right)\frac{\partial \vec{Q}1c}{\partial t} \\ & \quad + \Delta t \sigma\left(-\frac{h_r}{2}\right)\left(\frac{\partial \vec{Q}0n_r}{\partial t} - \frac{\partial \vec{Q}1n_r}{\partial t}\right) - \Delta t \phi\left(\frac{h_l}{2}\right)\left(\frac{\partial \vec{Q}0n_l}{\partial t} + \frac{\partial \vec{Q}1n_l}{\partial t}\right) = 0 \end{aligned}$$

$$\vec{R}0c = \frac{1}{h_c}\left(\vec{F}\left(\frac{h_c}{2}\right) - \vec{F}\left(-\frac{h_c}{2}\right)\right)$$

$$\begin{aligned} & \left[1 + \left(\phi\left(\frac{h_c}{2}\right) - \sigma\left(-\frac{h_c}{2}\right)\right)\frac{\Delta t}{h_c}\right]\frac{\partial \vec{Q}0c}{\partial t} + \left[\left(\phi\left(\frac{h_c}{2}\right) + \sigma\left(-\frac{h_c}{2}\right)\right)\frac{\Delta t}{h_c}\right]\frac{\partial \vec{Q}1c}{\partial t} = -\vec{R}0c \\ & \quad - \sigma\left(-\frac{h_r}{2}\right)\frac{\Delta t}{h_c}\left(\frac{\partial \vec{Q}0n_r}{\partial t} - \frac{\partial \vec{Q}1n_r}{\partial t}\right) + \phi\left(\frac{h_l}{2}\right)\frac{\Delta t}{h_c}\left(\frac{\partial \vec{Q}0n_l}{\partial t} + \frac{\partial \vec{Q}1n_l}{\partial t}\right) \end{aligned} \tag{6.5}$$

PDE for $\vec{Q}1$:

$$\frac{h_c}{3} \cdot \frac{\partial \vec{Q}1c}{\partial t} + \vec{F}\left(\frac{h_c}{2}\right)^{n+1} + \vec{F}\left(-\frac{h_c}{2}\right)^{n+1} = 2 \cdot \vec{F}0 \tag{6.6}$$

From equations (6.2), (6.3), and (6.4), we get

$$\begin{aligned} & \frac{h_c}{3} \cdot \frac{\partial \vec{Q}1c}{\partial t} + \vec{F}\left(\frac{h_c}{2}\right)^{n} + \vec{F}\left(-\frac{h_c}{2}\right)^{n} \\ & \quad + \Delta t\left(\phi\left(\frac{h_c}{2}\right) + \sigma\left(-\frac{h_c}{2}\right)\right)\frac{\partial \vec{Q}0c}{\partial t} + \Delta t\left(\phi\left(\frac{h_c}{2}\right) - \sigma\left(-\frac{h_c}{2}\right)\right)\frac{\partial \vec{Q}1c}{\partial t} \\ & \quad + \Delta t \sigma\left(-\frac{h_r}{2}\right)\left(\frac{\partial \vec{Q}0n_r}{\partial t} - \frac{\partial \vec{Q}1n_r}{\partial t}\right) + \Delta t \phi\left(\frac{h_l}{2}\right)\left(\frac{\partial \vec{Q}1n_l}{\partial t} + \frac{\partial \vec{Q}1n_l}{\partial t}\right) = 2 \cdot \vec{F}0 \end{aligned}$$

$$\vec{R}1c = \frac{3}{h_c}\left(\vec{F}\left(\frac{h_c}{2}\right) - \vec{F}\left(-\frac{h_c}{2}\right) - 2 \cdot \vec{F}0\right)$$

$$\left[\left(\phi\left(\frac{h_c}{2}\right)+\sigma\left(-\frac{h_c}{2}\right)\right)\frac{\Delta t}{h_c}\right]\frac{\partial \vec{Q}0c}{\partial t}+\left[1+\left(\phi\left(\frac{h_c}{2}\right)-\sigma\left(-\frac{h_c}{2}\right)\right)\frac{\Delta t}{h_c}\right]\frac{\partial \vec{Q}1c}{\partial t}=-\vec{R}1c$$
$$-\sigma\left(-\frac{h_r}{2}\right)\frac{\Delta t}{h_c}\left(\frac{\partial \vec{Q}0n_r}{\partial t}-\frac{\partial \vec{Q}1n_r}{\partial t}\right)-\phi\left(\frac{h_l}{2}\right)\frac{\Delta t}{h_c}\left(\frac{\partial \vec{Q}0n_l}{\partial t}+\frac{\partial \vec{Q}1n_l}{\partial t}\right) \qquad (6.7)$$

We employ a point implicit scheme to solve for $\frac{\partial \vec{Q}0}{\partial t}$ and $\frac{\partial \vec{Q}1}{\partial t}$ from equations 6.6 and 6.7. Point-implicit scheme is an iterative scheme in which the value from the previous iteration is used for all the terms in an equation except the term that corresponds to the dependent variable that is solved for. The convergence characteristics of such a scheme for numerical simulation of a quasi-1D flow in a convergent divergent nozzle is compared with the convergence characteristics of a four stage explicit Runge-Kutta time–marching scheme in fig. 12. The figure shows that an order of magnitude acceleration in convergence may be achieved using a point implicit scheme.

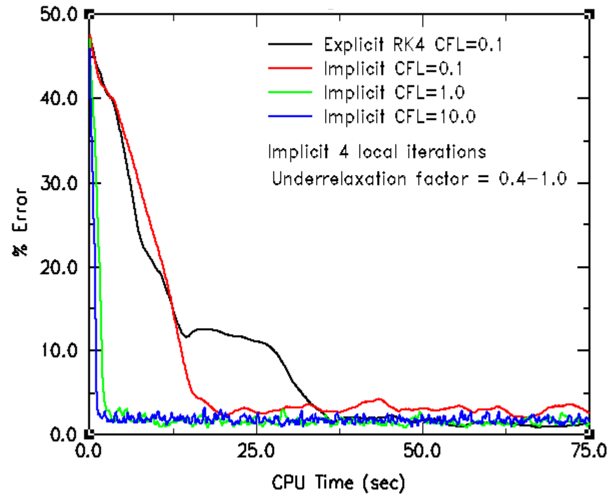


Figure 12. Convergence Characteristics of a p1 Point Implicit Scheme for Quasi-1D Flow in a Convergent-Divergent Nozzle

7.0 DG Formulation for the 3D Euler Equations

In this section, the formulation of the 3D DG Euler equations is presented. The structure of the equations follows the notation and conventions used in the 3D CEM DG code TEMPUS.

The 3D Euler equations can be written as

$$\frac{\partial \vec{Q}}{\partial t} + \vec{\nabla} \cdot \vec{F}\left(\vec{Q}\right) = 0 \ , \ \vec{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix} \tag{7.1}$$

$$\vec{F} = \left\langle \vec{f}_x, \vec{f}_y, \vec{f}_z \right\rangle, \ \vec{f}_x = \begin{bmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ \rho uw \\ (e+p)u \end{bmatrix}, \ \vec{f}_y = \begin{bmatrix} \rho v \\ \rho vu \\ p + \rho v^2 \\ \rho vw \\ (e+p)v \end{bmatrix}, \ \vec{f}_z = \begin{bmatrix} \rho w \\ \rho wu \\ \rho wv \\ p + \rho w^2 \\ (e+p)w \end{bmatrix} \tag{7.2}$$

where ρ is density, u is the velocity in the x-direction, v is the velocity in the y-direction, w is the velocity in the z-direction, e is the total energy per unit volume, and p is the pressure defined as

$p = (\gamma - 1)\left(e - \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)\right)$ where γ is the ratio of specific heats.

The DG form of the Euler equations can be written as

$$\int_{V_{ic}} \nu^i \frac{\partial \vec{Q}}{\partial t} dV = -\int_{S_{ic}} \nu^i \vec{F}\left(\vec{Q}\right) \cdot \hat{n} dS + \int_{V_{ic}} \nabla \nu^i \cdot \vec{F}\left(\vec{Q}\right) dV \tag{7.3}$$

where ic is the current cell, ν^i the ith basis function, V the cell volume, and S the cell face surface.

The basis functions in this version of the code are monomials. The monomials for up to P=2 (third order) are given in eqn. (7.4). For first order, P=0, i=0, and for second order, P=1, $0 \le i \le 3$.

$$\begin{aligned} &v_0 = 1 \\ &\nu_1 = x - x_c, \ \nu_2 = y - y_c, \ \nu_3 = z - z_c \\ &v_4 = (x - x_c)\cdot(y - y_c), \ \nu_5 = (x - x_c)\cdot(z - z_c), \ \nu_6 = (y - y_c)\cdot(z - z_c) \\ &\nu_7 = (x - x_c)^2, \ \nu_8 = (y - y_c)^2, \ \nu_9 = (z - z_c)^2 \end{aligned} \tag{7.4}$$

where c is for the cell centroid.

The solution vector Q is expanded in the basis functions and the component j of the vector Q is

$$Q_j = \sum_{i=0}^{JP} \nu^i q_j^i \tag{7.5}$$

with JP=0 for P=0, JP=3 for P=1, and JP=9 for P=2.

Similarly, the flux vector F is expanded in a Taylor series about the cell centroid. The expansion for the j^{th} component of the flux vector may be written as

$$F_j = \sum_{i=0}^{JP} \nu^i f_j^i \tag{7.6}$$

with

$$f_j^0 = F_j\left(\vec{q}^{\,0}\right) \text{ for i=0 (P=0)} \tag{7.7}$$

$$f_j^i = \sum_{k=1}^{5} \frac{\partial F_j}{\partial Q_k} q_k^i \text{ for i=1,3 (P=1)}$$

$$f_j^i = \sum_{k=1}^{5}\sum_{l=1}^{5} \frac{\partial^2 F_j}{\partial Q_k \partial Q_l} q_k^{i1} q_l^{i2} + \sum_{k=1}^{5} \frac{\partial F_j}{\partial Q_k} q_k^i \quad \text{for i=4,9 (P=2)}$$

for i=4, i1=1, i2=2, for i=5, i1=1, i2=3, for i=6, i1=2, i2=3, for i=7, i1=1, i2=1, for i=8, i1=2, i2=2, and for i=9, i1=3, i2=3.

8.0 Evaluation of Integrals in eqn. 7.3

This section describes how each of the integral terms in eqn. (7.3) is evaluated. The first integral to be evaluated is $\int\limits_{V_{ic}} \nu^i \frac{\partial \vec{Q}}{\partial t} dV$ which will be rewritten using the basis function expansion for Q.

$$\int\limits_{V_{ic}} \nu^i \frac{\partial \vec{Q}}{\partial t} dV = \int\limits_{V_{ic}} \nu^i \sum_{j=0}^{JP} \nu^j \frac{\partial q^j}{\partial t} = \sum_{j=0}^{JP} \int\limits_{V_{ic}} \nu^i \nu^j \frac{\partial q^j}{\partial t} dV$$

with $dV = \left|\frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)}\right| \partial\xi\partial\eta\partial\zeta$, $\int\limits_{V_{ic}} \nu^i \frac{\partial \vec{Q}}{\partial t} dV$ can be rewritten in transformed coordinates as

$$\int\limits_{V_{ic}} \nu^i \frac{\partial \vec{Q}}{\partial t} dV = \sum_{j=0}^{JP} \frac{\partial q^j}{\partial t} \int\limits_{\xi=-1}^{+1} \int\limits_{\eta=-1}^{+1} \int\limits_{\zeta=-1}^{+1} \nu^i \nu^j \left|\frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)}\right| \partial\xi\partial\eta\partial\zeta = \sum_{j=0}^{JP} m_{ij} \frac{\partial q^j}{\partial t} \tag{8.1}$$

$M = \lfloor m_{ij} \rfloor$ is the mass matrix.

The next integral to be evaluated is $\int\limits_{V_{ic}} \nabla \nu^i \cdot \vec{F}(\vec{Q}) dV$.

$$\int\limits_{V_{ic}} \nabla \nu^i \cdot \vec{F}(\vec{Q}) dV = \int\limits_{V_{ic}} \left(\frac{\partial \nu^i}{\partial x} f_x + \frac{\partial \nu^i}{\partial y} f_y + \frac{\partial \nu^i}{\partial z} f_z \right) dV$$

Following the evaluation process of $\int\limits_{V_{ic}} \nu^i \frac{\partial \vec{Q}}{\partial t} dV$, $\int\limits_{V_{ic}} \nabla \nu^i \cdot \vec{F}(\vec{Q}) dV$ is rewritten using the basis function expansion of f_x, f_y, and f_z as

$$\int\limits_{V_{ic}} \nabla \nu^i \cdot \vec{F}(\vec{Q}) dV = \int\limits_{V_{ic}} \sum_{j=0}^{JP} \left(\frac{\partial \nu^i}{\partial x} \nu^j f_x^j + \frac{\partial \nu^i}{\partial y} \nu^j f_y^j + \frac{\partial \nu^i}{\partial z} \nu^j f_z^j \right) dV$$

Finally using the expression for dV given earlier, we get

$$\int\limits_{V_{ic}} \nabla \nu^i \cdot \vec{F}(\vec{Q}) dV =$$

$$\sum_{j=0}^{JP} f_x^j \int\limits_{\xi=-1}^{+1} \int\limits_{\eta=-1}^{+1} \int\limits_{\zeta=-1}^{+1} \frac{\partial \nu^i}{\partial x} \nu^j \left| \frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)} \right| \partial\xi \partial\eta \partial\zeta + \sum_{j=0}^{JP} f_y^j \int\limits_{\xi=-1}^{+1} \int\limits_{\eta=-1}^{+1} \int\limits_{\zeta=-1}^{+1} \frac{\partial \nu^i}{\partial y} \nu^j \left| \frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)} \right| \partial\xi \partial\eta \partial\zeta$$
$$+ \sum_{j=0}^{JP} f_z^j \int\limits_{\xi=-1}^{+1} \int\limits_{\eta=-1}^{+1} \int\limits_{\zeta=-1}^{+1} \frac{\partial \nu^i}{\partial z} \nu^j \left| \frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)} \right| \partial\xi \partial\eta \partial\zeta$$

or

$$\int\limits_{V_{ic}} \nabla \nu^i \cdot \vec{F}(\vec{Q}) dV = \sum_{j=0}^{JP} \left(mx_{ij} f_x^j + my_{ij} f_y^j + mz_{ij} f_z^j \right) \tag{8.2}$$

where $Mx, My, Mz = \lfloor mx_{ij} \rfloor, \lfloor my_{ij} \rfloor, \lfloor mz_{ij} \rfloor$ are the gradient mass matrices.

The final integral is $\int\limits_{S_{ic}} \nu^i \vec{F}(\vec{Q}) \cdot \hat{n} dS$. This is a surface integral at the interior cell faces which will use information from the current cell (ic) and the neighbor cell (inb). The face flux is formed using the local Lax-Friedrichs scheme.

$$\vec{F}(\vec{Q}) \cdot \hat{n} = \frac{1}{2}\left(\vec{F}_{ic} + \vec{F}_{inb}\right) + \frac{1}{2}\lambda_{\max}\left(\vec{Q}_{ic} - \vec{Q}_{inb}\right)$$

where $\lambda_{\max} = |\hat{u} + \hat{c}|$, $\hat{u} = u \cdot n_x + v \cdot n_y + w \cdot n_z$ and $\hat{c} = \sqrt{\frac{\gamma p}{\rho}}$.

$\hat{n} = \langle n_x, n_y, n_z \rangle$ is the unit normal at the cell face and

$\vec{F}_{ic} = \vec{f}_x \cdot \hat{n}_x + \vec{f}_y \cdot \hat{n}_y + \vec{f}_z \cdot \hat{n}_z$ (all expansions from cell centroid ic)

$\vec{F}_{inb} = \vec{f}_x \cdot \hat{n}_x + \vec{f}_y \cdot \hat{n}_y + \vec{f}_z \cdot \hat{n}_z$ (all expansions from cell centroid inb)

Expand $\vec{F}$ and $\vec{Q}$ in the basis functions and $\int_{S_{ic}} \nu^i \vec{F}(\vec{Q}) \cdot \hat{n} dS$ can be rewritten using the following expression for $dS = \left| \frac{\partial(x,y,z)}{\partial\xi} \times \frac{\partial(x,y,z)}{\partial\eta} \right| \partial\xi\partial\eta$

$$\begin{aligned}
\int_{S_{ic}} \nu^i \vec{F}(\vec{Q}) \cdot \hat{n} dS &= \sum_{j=0}^{JP} \frac{1}{2} \left(f_{ic}^j \int_{\xi=-1}^{+1} \int_{\eta=-1}^{+1} \nu_{ic}^i \nu_{ic}^j \left| \frac{\partial(x,y,z)}{\partial\xi} \times \frac{\partial(x,y,z)}{\partial\eta} \right| \partial\xi\partial\eta \right) \\
&+ \sum_{j=0}^{JP} \frac{1}{2} \left(f_{inb}^j \int_{\xi=-1}^{+1} \int_{\eta=-1}^{+1} \nu_{ic}^i \nu_{inb}^j \left| \frac{\partial(x,y,z)}{\partial\xi} \times \frac{\partial(x,y,z)}{\partial\eta} \right| \partial\xi\partial\eta \right) \\
&+ \sum_{j=0}^{JP} \frac{1}{2} \lambda_{\max} \left(q_{ic}^j \int_{\xi=-1}^{+1} \int_{\eta=-1}^{+1} \nu_{ic}^i \nu_{ic}^j \left| \frac{\partial(x,y,z)}{\partial\xi} \times \frac{\partial(x,y,z)}{\partial\eta} \right| \partial\xi\partial\eta \right) \\
&- \sum_{j=0}^{JP} \frac{1}{2} \lambda_{\max} \left(q_{inb}^j \int_{\xi=-1}^{+1} \int_{\eta=-1}^{+1} \nu_{ic}^i \nu_{inb}^j \left| \frac{\partial(x,y,z)}{\partial\xi} \times \frac{\partial(x,y,z)}{\partial\eta} \right| \partial\xi\partial\eta \right)
\end{aligned}$$

Finally, $\int_{S_{ic}} \nu^i \vec{F}(\vec{Q}) \cdot \hat{n} dS$ can be rewritten as

$$\int_{S_{ic}} \nu^i \vec{F}(\vec{Q}) \cdot \hat{n} dS = \frac{1}{2} \sum_{j=0}^{JP} \left(s_{ij} f_{ic}^j + f_{ij} f_{inb}^j + \lambda_{\max} \left(s_{ij} q_{ic}^j - f_{ij} q_{inb}^j \right) \right) \tag{8.3}$$

where $S = \lfloor s_{ij} \rfloor$ is the surface matrix and $F = \lfloor f_{ij} \rfloor$ is the face matrix.

At boundaries, the flux computation follows the same structure except that now $\vec{Q}_{inb} = function(\vec{Q}_{ic})$ with the functional dependency determined by the boundary condition employed. Currently inviscid surface, supersonic inflow, and a combination of subsonic inflow/outflow boundary conditions have been implemented.

Now that all the integrals have been evaluated, the 3D DG Euler equations, eqn. (7.3), can be written as

$$M \frac{\partial \vec{Q}_{ic}}{\partial t} = -\frac{1}{2} \left(S\vec{F}_{ic} + F\vec{F}_{inb} \right) - \frac{\lambda_{\max}}{2} \left(S\vec{Q}_{ic} + F\vec{Q}_{inb} \right) + \left(Mx \cdot \vec{f}_x + My \cdot \vec{f}_y + Mz \cdot \vec{f}_z \right)_{ic} \tag{8.4}$$

All the integrals in M, S, F, Mx, My, and Mz are evaluated using Gaussian quadrature. The solution is advanced in time using a 4 stage Runge-Kutta time algorithm.

9.0 Conversion of CEM Solver to Euler Flow Solver

The structure of the CEM solver is presented in the form of a flowchart in Appendix 1. As was stated in our proposal, it has been possible to convert the CEM solver to an Euler solver without making any appreciable change to the structure of the solver. The details of the Euler solver are presented in the following paragraphs.

Basis function/Geometry Data Base
The subroutines *hex_vol_moms.c, hex_sur_moms.c,* and *hex_fac_moms.c* are used in original form to compute the M, Mx, My, Mz, S, and F matrices which will be used in solving eqn (8.4).

Flow initialization
The initialization subroutine *q_init.f* has been modified to initialize the flow field using a constant freestream flow condition defined by the user in the form of the variables *uincq**n**, **n** =1,6*. This initialization needs to be recoded later for more general flow initialization.

Flux computation
The flux subroutine *iflxn.f* was modified to compute the Euler fluxes using eqn (8.3). This subroutine is modified only slightly from the CEM version. In CEM, the fluxes f_{ic} and f_{inb} are linear combinations of q_{ic} and q_{inb}. In the case of inviscid flow, eqns. (7.6) and (7.7) need to be used since the fluxes are nonlinear. Once the flux coefficients f^j are computed using eqns. (7.6) and (7.7), the original structure of the subroutine can be retained since the flux subroutine was developed using the linear combination of the solution coefficients q^j as the flux. The S and F matrices in this subroutine are used without change from the original subroutine. Subroutines *jacobian.f* and *fluxcoefficient.f* were added to compute the flux coefficients.

Boundary conditions
In this version, boundary condition subroutine *ubc_pc.f* has been modified to be a inviscid surface tangency boundary condition, *ubc_rc.f* has been modified to be a plane of symmetry boundary condition, and *ubc_ob.f* has been modified to be a subsonic/supersonic inflow/outflow freestream boundary condition. New subroutines *hex_ob_quads.c* and *hex_rc_quads.c* were added (essentially just a copy of *hex_pc_quads.c*) with the appropriate “ob” or “rc” variables dynamically allocated and used. These new subroutines allow these boundary condition subroutines to use the same data base that the “pc” subroutine is using. A more complete boundary condition library needs to be developed since a general flow code needs many more types of boundary conditions.

Gradient Mass Term
The subroutine *ivlux.f* needs to be modifed for the Euler fluxes (again since in CEM the fluxes are a linear combination of Q). This subroutine computes the gradient mass integral term of eqn. (8.2). Again where solution coefficients q^j were being used for the flux, f^j flux coefficients are

computed and used. This is just the same procedure as was done in modifying the flux computations.

Post-processing
Finally the output subroutine *write_soln.f* needs to be modifed to output the appropriate flow variables for the Euler flow solver and remove all CEM output.

In the conversion, one can see that the modularity of the CEM TEMPUS code allows most of the code to be used in the original state. Only four segments of the code needed to be modified, namely, (1) flow initialization; (2) flux computation; (3) boundary conditions; and (4) post-processing.

10.0 Verification of the Implementation

In this section, we present results obtained using the 3D DG Euler solver for inviscid flow over a 2D cylinder. The grid employed in the simulation is shown in fig. 13. The geometry considered is a half cylinder with radius =1.0, and span width = 10.0. The outer boundary is a half circle of radius = 30.0. The grid has 30 points equally spaced on the circumference of the cylinder and 30 points normal to the cylinder surface with the first point off the cylinder surface at a distance = 0.05. Four equally spaced span points are used. The cell topology is hexagonal. This grid was run using the 3D DG Euler code and the fully validated structured grid solver, USA [12]. The flow conditions employed are

$\rho = 1.0, \quad p = 1.0, \quad u = 0.3, \quad v = 0.0, \quad w = 0.0, \quad \gamma = 1.4$

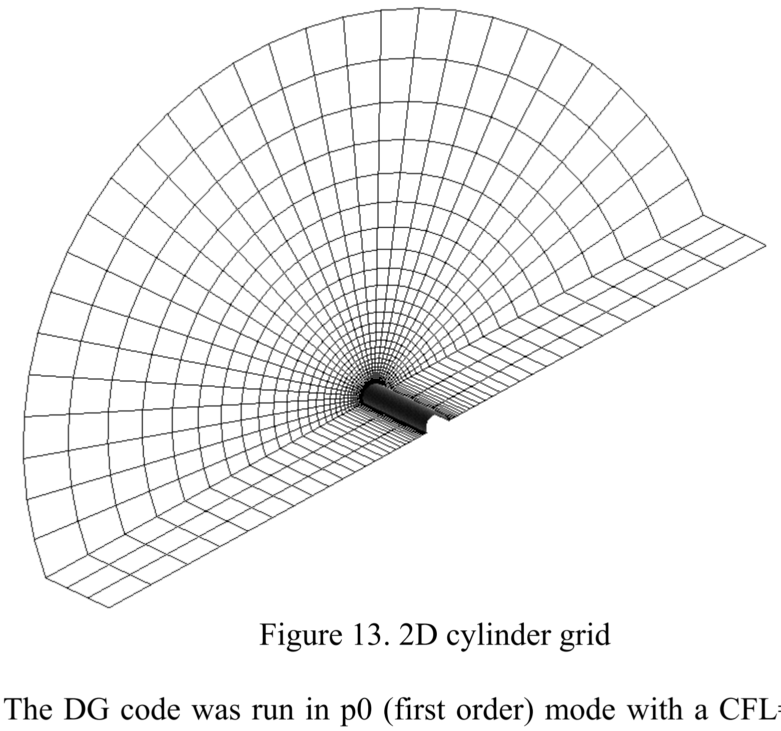


Figure 13. 2D cylinder grid

The DG code was run in p0 (first order) mode with a CFL=1.0 and in p1 (second order) mode with CFL=0.3. Fig. 14 shows that the DG code is slightly less dissipative than the USA code for the same order of accuracy. The potential flow solution is shown as a reference. One must remember that in solving the Euler equations, there will be numerical dissipation which will cause differences from the potential flow solution. Further testing of the Euler DG code needs to be done. Validation of the p2 (third order) mode will be carried out in Phase II.

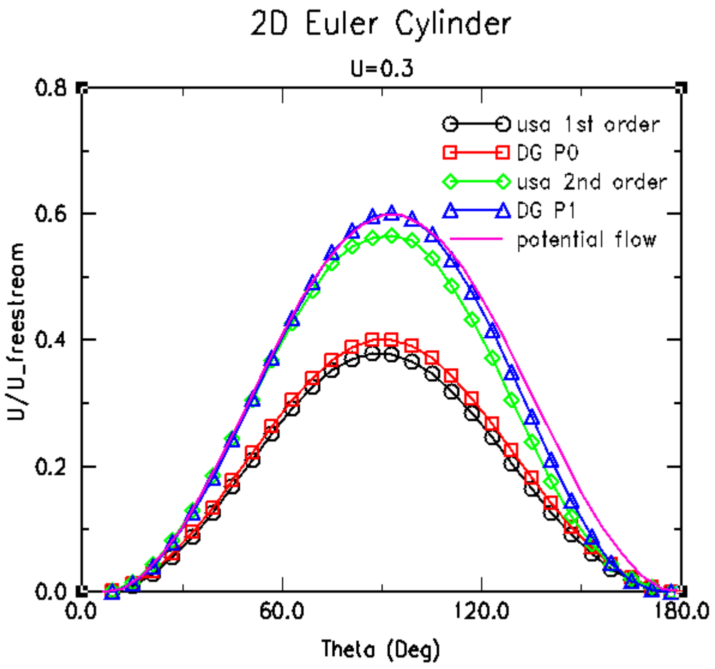


Figure 14. u velocity on cylinder surface

11.0 Design of a Common Platform for Multi-physics Computation

In this section, we discuss the design of a common platform for multi-physics computation.

11.1 Multidisciplinary analysis

In a multidisciplinary analysis, from the point of view of numerical simulation, following situations may arise:

1. Different physical phenomena involved in the analysis may require different computational domains (Fig. 15). For example, in the case of aeroelasticity the computational domains for the equations of fluid flow and elasticity are different. The two phenomena interact at the boundaries of the domain.
2. The physical phenomena involved may require the same computational domain (Fig. 16). In this case there are two different possibilities:
    a. The same grid may be employed in solving all the governing equations.
    b. All physical phenomena involved may not require the same grid resolution and as such different grids may be used for different physical phenomena.
3. A combination of situations 1 and 2.

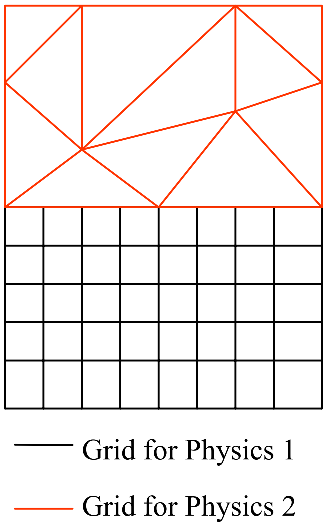


Figure 15. Different computational domains for different physics

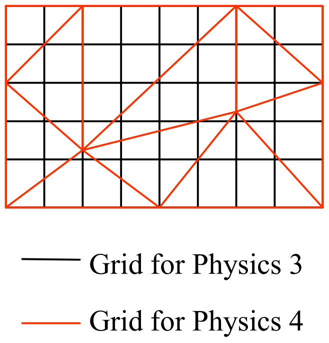


Figure 16. Same computational domains for different physics

In all these situations the governing equations may either be solved in a coupled manner or in an uncoupled manner. A coupled solver would involve inversion of a single matrix while an uncoupled solver would involve inverting several matrices. In either case a highly accurate interpolation module is required when the volume grids in the same computational domain or surface grids in the case of different computational domains are different. The DG scheme makes such interpolations quite simple. The expression for the expansion of the dependent variables in terms of the basis functions also serves the role of interpolation function. The TEMPUS code has a modular structure to be able to be modified to handle the three situations listed above.

The partial differential equations that govern different physical phenomena may either be hyperbolic, parabolic or elliptic. From a numerical simulation point of view they may be categorized as either in a conservation form or in a non-conservation form. In the case of equations in conservation form with all positive eigen-values, the solution process mainly involves computation of flux vectors at element boundaries, volume source terms, and in the case of implicit schemes, the flux Jacobian. Therefore, it is indeed possible, as has been demonstrated by us in the case of computational electromagnetics (CEM) and computational fluid dynamics (CFD), that a common tool may be developed with individual modules for computation of flux terms, source terms and flux Jacobian to account for different physical phenomena involved.

The discussion above leads to the following ingredients for a multidisciplinary tool:

1. A module for establishing connectivity at the boundaries of different computational domains (such as solid/liquid interface).
2. A routine to compute connectivity between different grid cells in the same computational domain but belonging to different grids.
3. A solver for equations in conservation form with only positive eigen-values that includes individual flux, source, and Jacobian modules for various physical phenomena involved in the simulation.
4. Individual solvers for all other governing equations.
5. Interface for exchanging information at computational boundaries for different physics.

In addition to having the ingredients listed above, a successful commercial multidisciplinary simulation tool should be complete, user friendly, fast, and should require reasonable computing resources. Completeness requires the tool to perform all the tasks related to the simulation. User friendliness is obtainable through the development of a graphical user interface (GUI) that is intuitive and that guides the user through the process quite smoothly. Speed and resource requirements are closely associated with the numerical scheme employed in the simulation and the extent of intelligence built into the tool.

We will use HyPerComp's CEM simulation tool TEMPUS as the initial framework to develop this multi-disciplinary/multi-physics tool. The 3D DG CEM solver part of TEMPUS has been mentioned previously in the report as it is the solver which was used in the conversion to a 3D DG Euler solver. Now the whole TEMPUS environment will be described.

11.2 TEMPUS

In the last three years, HyPerComp has been involved in the development of TEMPUS, a complete tool for the simulation of Maxwell's equations. Starting with the creation of the geometry of the system to be simulated, TEMPUS has the ability to repair geometric imperfections, define a computational domain and discretize it, setup boundary conditions, prepare input required by the solver, perform domain decomposition for parallel processing, and execute the solver. Our design of a common platform for multidisciplinary analysis is based on our experience in developing TEMPUS.

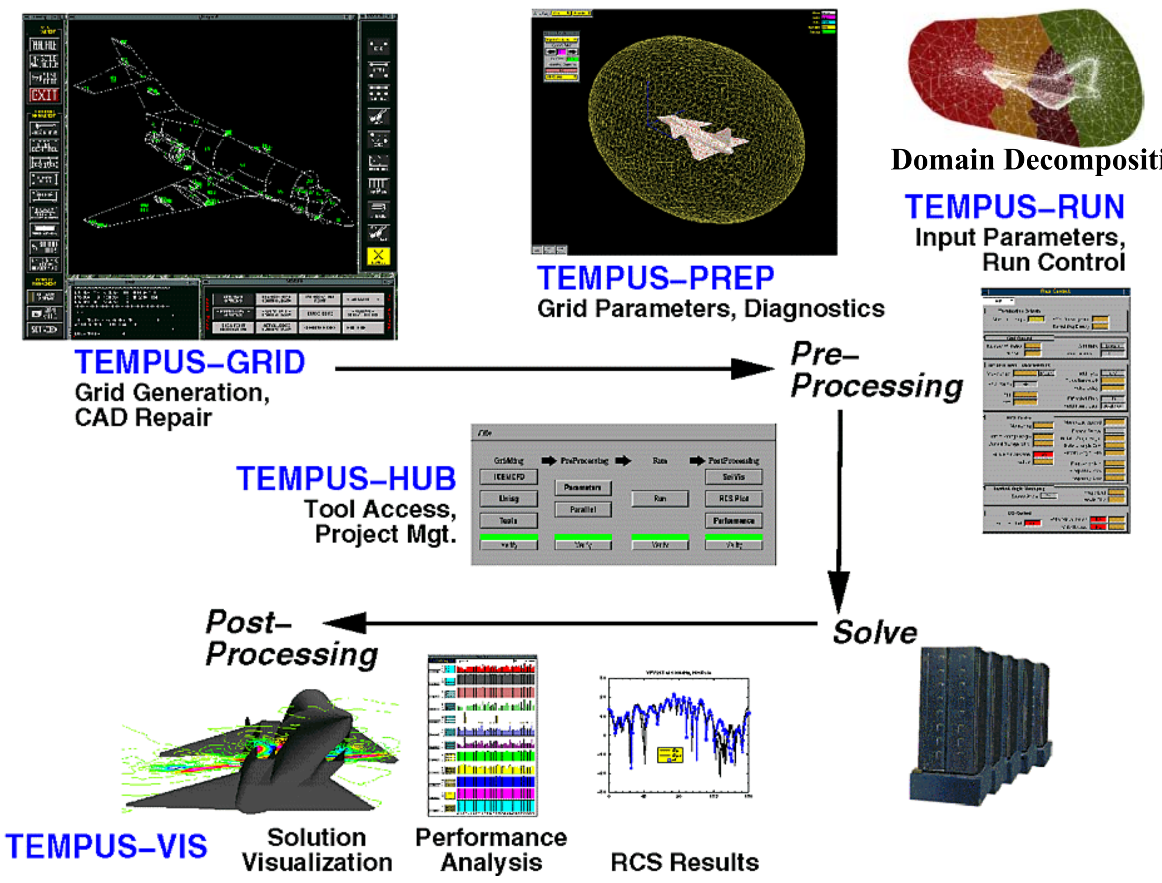


Figure 17. TEMPUS environment for wide band CEM applications

At the core of the TEMPUS environment, there is a collection of technologies that represent the strength of TEMPUS. Some of them are listed here:

1. A higher order discontinuous Galerkin method that is currently operational up to 10th order for spatial representation of electric and magnetic fields inside each computational finite-volume cell.
2. A hybrid unstructured grid framework for modeling complete targets including engine inlets with stages of fan blades
3. Modeling of general materials such as lossy/lossless dielectric, resistive cards, impedance layers and dispersive media
4. Highly scalable parallel code architecture that is transportable across different platforms.

11.3 Higher Order Schemes for Multidisciplinary Analysis

Our choice for higher-order multidisciplinary analysis is the DG scheme. This choice is based not only on our extraordinary success in developing a DG based higher-order solver for CEM that clearly demonstrates the ability of a higher-order DG scheme to produce very accurate solutions with appreciable decrease in turnaround time, but also on the fact that DG is the most appropriate choice for multidisciplinary analysis based on the discussion presented in the next three paragraphs.

One of the earliest attempts at developing highly accurate numerical schemes employed a spectral method. In a spectral method the dependent variables $\vec{Q}$ are expanded in terms of a basis function $\left(v^i(x,y,z), i=1,n\right)$ as $\vec{Q} = \sum_{i=1}^{n} \vec{Q}^i v^i$. The solution process solves for the coefficients $\vec{Q}^i i=1,n$. This scheme requires the solution to be continuous across the computational domain and performs exceedingly well for problems that satisfy the requirement. Unfortunately a large category of problems that we encounter in real life does not satisfy such a requirement. A natural extension to the spectral method that is less restrictive is to employ a local basis function rather than a global one. The DG scheme does just that. Use of a local basis function permits the solution to be discontinuous at element boundaries and thus enables application of the scheme to problems wherein accurate solutions may be obtained by resolving discontinuities at element interfaces. Within each element the solution is assumed to be continuous. This restriction is obviously much less severe than the restriction imposed by a spectral method wherein the solution is required to be continuous across the whole computational domain.

Essentially non-oscillatory (ENO) schemes and the weighted ENO (WENO) schemes also employ local basis functions. The main differences between the ENO schemes and the DG schemes are:

1. The ENO schemes solve for the dependent variables while the DG schemes solve for the coefficients $\vec{Q}^i i=1,n$
2. In an ENO scheme the coefficients $\vec{Q}^i i=1,n$ are computed by an appropriate fit of the data in the neighborhood of an element while in a DG scheme a weak form of the solution is employed in the computation of the coefficients.

Item 1 above implies that the DG scheme is solving for more unknowns than the ENO schemes. Item 2 implies that the ENO schemes employ a much larger numerical stencil than the DG schemes. As advantageous as it is to solve for less number of unknowns, the use of a much larger stencil in ENO schemes renders the schemes almost unusable due to severe stability limitations.

One of the attractive features of the DG schemes is the compactness of the stencil employed. That is, the equations that determine the coefficients $\vec{Q}^i i=1,n$ depend only on the data in the immediate neighborhood of a stencil. The PADE type of finite difference schemes also has this property of compactness. Unfortunately such schemes do not extend easily to unstructured grids. The DG schemes extend quite naturally to discretizing the computational domain using elements of arbitrary shapes and as such have a clear advantage over the PADE schemes. The advantages that unstructured grids have over structured grids are so powerful that it is almost imperative that we choose a scheme that extends easily to unstructured grids.

Based on the above discussion, the higher-order multidisciplinary platform that we will develop will employ a DG scheme and unstructured grids.

11.4 Computational Environment for Multidisciplinary Analysis

There are two principal methods to integrate multiple physical phenomena into a single computational environment. They are:

(a) Tight (monolithic) coupling, in which the various physical modules are integrated into one large matrix system using a single numerical strategy and inverted in bulk, and,

(b) Loose (iterative) coupling, in which the various physical models are solved individually and global iterations are performed in order to converge them in a given time step

Most of the multidisciplinary analyses currently carried out employ loosely coupled algorithms. For example, in the analysis of the interaction between aerodynamic and elastic forces, aerodynamic loads are computed first and the resulting elastic response is then evaluated to obtain a modified shape which is used to recompute the aerodynamic loads. This process is continued until a convergence criterion is satisfied. As such a loosely coupled algorithm only requires an interface between different solvers employed in the simulation of different physical phenomena. This may be quite easily achieved by developing a modular environment and the required interfaces. Our goal is to go beyond the development of such an environment and develop strongly coupled algorithms wherever possible. Magnetogasdynamics, multiphase, and free surface flows are examples wherein a strongly coupled algorithm may be developed. Fig. 18 shows a sampling of multidisciplinary problems that may be handled by the proposed environment.

Although a loosely coupled multidisciplinary analysis may appear attractive for its simplicity in implementation, freedom in discretization procedures, and ease in extending to different phenomena, a tightly coupled analysis is certainly desirable since it is likely to be more accurate, more stable, and will permit use of larger time-steps for time-accurate simulations. Our choice is a combination of two. We propose to employ a tightly coupled strategy wherever possible. For example, a multi-species flow with chemical reaction may be solved in a coupled manner while the analysis of the interaction between aerodynamic and structural forces will require a loosely coupled strategy.

One of the challenges that we need to address is related to optimizing the resource requirements. In a higher-order DG scheme, the number of dependent variables increases with the order of the scheme. Since the computing resources required are usually nonlinear functions of the number of dependent variables, it is quite easy to stress the available resources beyond their limits. The main task is to ensure that we employ an adaptive order scheme so that we use only as high an order of scheme for an element as is required by the solution and not any higher.

The second challenge is developing an efficient strategy for parallel execution of multidisciplinary simulation. In the case of a tightly coupled strategy, this is not a major issue since the same grid is employed in solving different governing equations. Any standard domain decomposition module such as METIS may be employed to ensure efficient parallel performance. In the case of a loosely coupled system, we first need to evaluate the CPU time required by different solvers and based on the grid employed and the time required by the solver we need to develop an optimal distribution of work among available processors. In order to ensure maximum efficiency, we may have to perform different number of iterations for different solvers before data is exchanged between them.

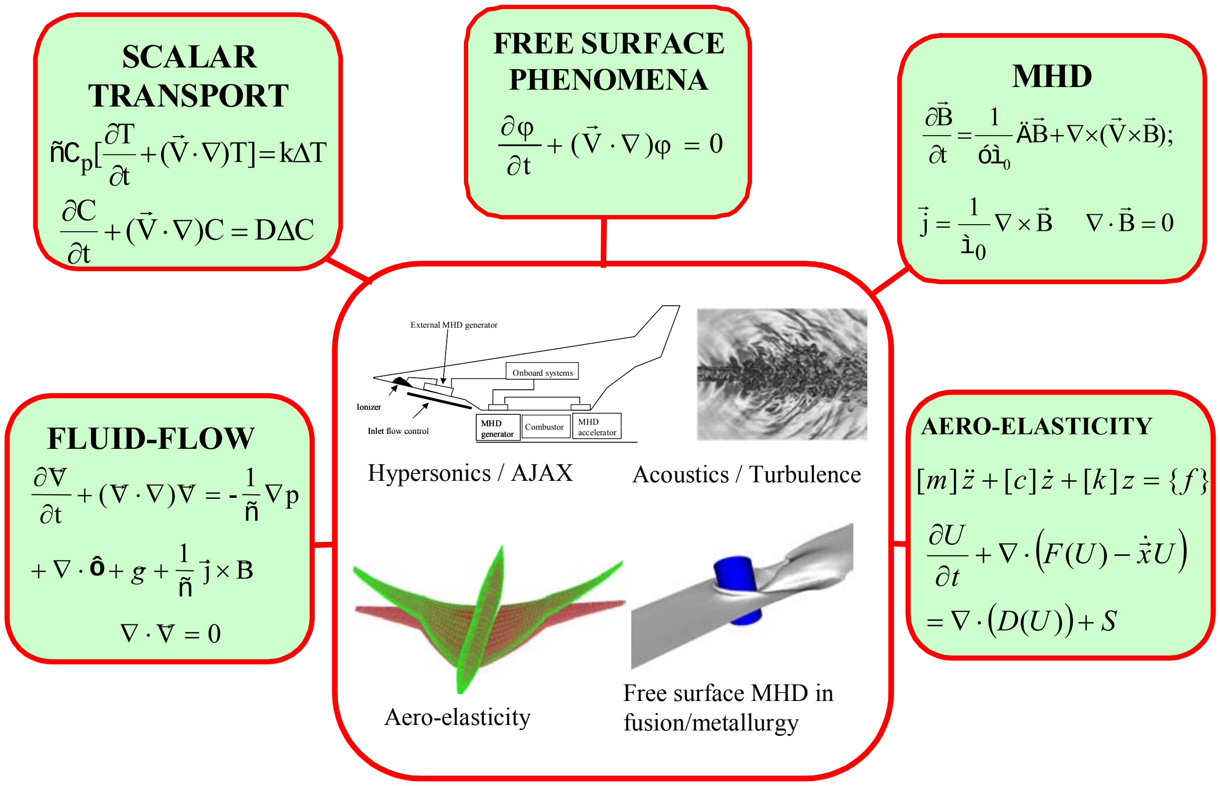
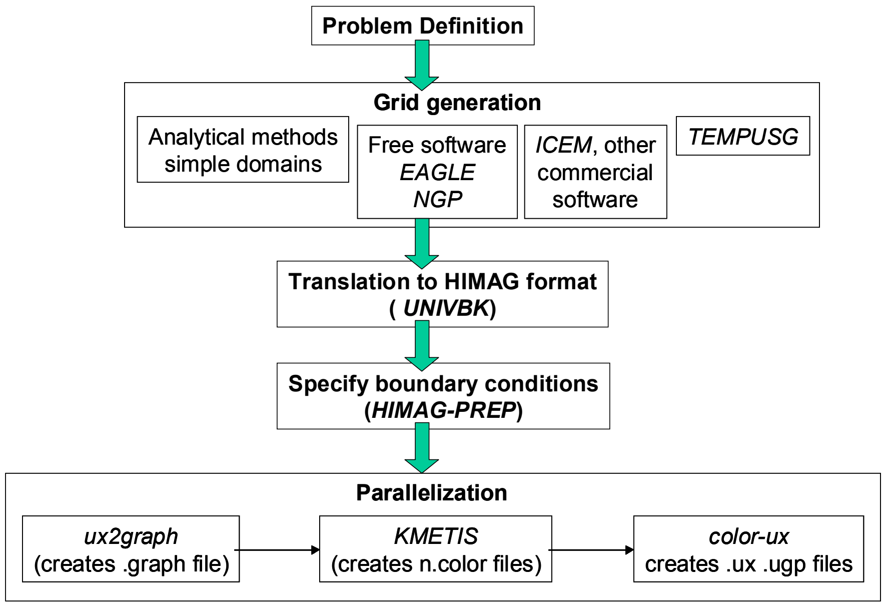

SCALAR TRANSPORT
$ñC_p[\frac{\partial T}{\partial t}+(\vec{V}\cdot\nabla)T]=k\Delta T$
$\frac{\partial C}{\partial t}+(\vec{V}\cdot\nabla)C=D\Delta C$
FREE SURFACE PHENOMENA
$\frac{\partial \varphi}{\partial t}+(\vec{V}\cdot\nabla)\varphi=0$
MHD
$\frac{\partial \vec{B}}{\partial t}=\frac{1}{óì_0}Ä\vec{B}+\nabla\times(\vec{V}\times\vec{B});$
$\vec{j}=\frac{1}{ì_0}\nabla\times\vec{B} \quad \nabla\cdot\vec{B}=0$
FLUID-FLOW
$\frac{\partial \vec{V}}{\partial t}+(\vec{V}\cdot\nabla)\vec{V}=-\frac{1}{ñ}\nabla p$
$+\nabla\cdot ô+g+\frac{1}{ñ}\vec{j}\times B$
$\nabla\cdot\vec{V}=0$
AERO-ELASTICITY
$[m]\ddot{z}+[c]\dot{z}+[k]z=\{f\}$
$\frac{\partial U}{\partial t}+\nabla\cdot\left(F(U)-\dot{x}U\right)$
$=\nabla\cdot\left(D(U)\right)+S$
External MHD generator
Onboard systems
Ionizer
Inlet flow control
MHD generator
Combustor
MHD accelerator
Hypersonics / AJAX
Acoustics / Turbulence
Aero-elasticity
Free surface MHD in fusion/metallurgy

Figure 18. Some Multiphysics Problems of Interest

Figs. 19 and 20 show a sample strategy to parallelize an incompressible free surface MHD flow simulation. Various physical elements are individually converged to an acceptable tolerance. Different physical phenomena may require different grid resolution, thus requiring a strategy to exchange information across grid levels. As discussed in section 11.1, use of DG as a higher-order scheme simplifies exchange of information between different grids and the only module that needs to be developed is an octree based technique for locating the element in one grid that contain a given point from another grid.



Problem Definition
Grid generation
Analytical methods simple domains
Free software EAGLE NGP
ICEM, other commercial software
TEMPUSG
Translation to HIMAG format (UNIVBK)
Specify boundary conditions (HIMAG-PREP)
Parallelization
ux2graph (creates .graph file)
KMETIS (creates n.color files)
color-ux creates .ux .ugp files

Figure 19. Pre-processing setup for HIMAG: **H**yPerComp **I**ncompressible **M**HD solver for **A**rbitrary **G**eometries

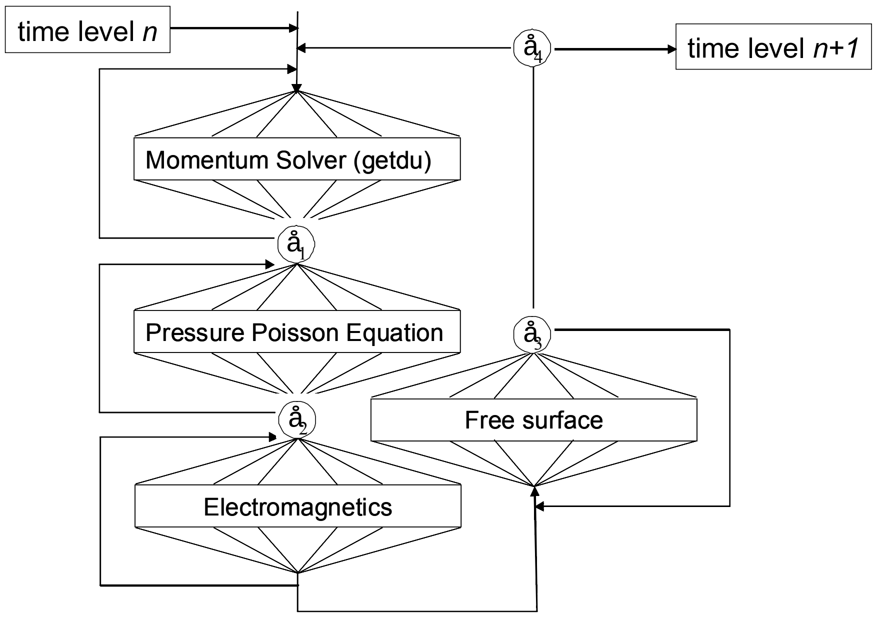


Figure 20. Parallelization strategy for the incompressible MHD solver HIMAG – the ε s represent tolerances of individual solvers

12.0 Suggestions for future work

Our proposal for the continuation of the work done under Phase I involves the following 3 categories:

1. Construction of a general purpose three-dimensional code environment for multiphysics simulation. The starting point for this task is the existing TEMPUS CEM environment.
2. Implementation and enhancement of high order algorithms addressing multiphysics
3. Demonstration of the high order platform for selected candidate problems requiring multiphysics simulations

Detailed list of subtasks to be done under each of the tasks listed above is given below.

T-1: Multidisciplinary Environment Development Tasks

T-1.1: Starting with the TEMPUS CEM environment, develop and incorporate data structure for modeling multiphysics simulations and optimize memory resource requirements
T-1.2: Develop GUIs for problem set up, boundary conditions and run set up for various physics
T-1.3: Develop interfaces including interpolation routines for data communications between different solvers for different pysics.
T-1.4: Develop appropriate parallel MPI routines for multiphysics data communication across nodal boundaries
- 1.4.1 Parallel data structure for moving mesh
- 1.4.2 Load balancing for moving/adaptive mesh

T-1.5: Add to the GUI a database for relevant physical properties
T-1.6: Develop makefile/compiler options for portability across different parallel hardware

T-2: High Order Algorithm Enhancement Tasks

T-2.1: Develop orthonormal basis function set for at least up to 10th order for multiphysics
T-2.2: Special basis functions for singular regions (e.g. sharp leading edge regions)
T-2.3: High order boundary condition procedures (e.g. nonreflecting outer boundary conditions)
T-2.4: Develop linearization procedures for advection terms
T-2.5: Develop Riemann flux routines for different physics
T-2.6: Multi-order and implicit time step procedures
T-2.7: Limiters/filters/pre-conditioners/penalty functions and other stabilizing/code acceleration techniques
T-2.8: Incorporation of high order geometry representation and nonorthogonal grid effects
T-2.9: Coupling procedures for elliptic, parabolic and hyperbolic regions/aspects of multiphysics
T-2.10:High order level set procedures for free surface tracking

T-3: Tasks on Multiphysics Simulation/Demonstration

This set of tasks will complement ongoing research efforts and will synchronize with the time lines in which model development from those projects will supplement and validate present results. Unit problems in MHD, turbulent fluid flow , acoustics and aerothermal flows will be considered. The first two problems below will be used as demonstration cases on large scale parallel LINUX cluster. Specifics of the later tasks will be determined jointly with the program monitor.

T-3.1: Euler/Navier-Stokes/Maxwell equations for MGD with chemical kinetics
Hypersonic inlet flow with viscosity, (RANS) turbulence and MHD terms.
T-3.2: Incompressible flow with MHD and a free surface
Free surface flow of liquid metal past a cylinder in the presence of a magnetic field
T-3.3: Demonstration of high order DG for a candidate turbulent flow problem
T-3.4: Demonstration of a flexible structure with active feedback control
T-3.5: Study of high order absorbing outer boundary conditions for an acoustics problem
T-3.6: Application of high order DG for accurate simulation of aerothermal prediction

T-4: Analysis
In addition to all of the above, we intend to perform preliminary analysis of the DG scheme as we develop it, and study its stability and convergence properties.

13.0 References

[1] Shankar, V., "CFD-Based Higher order methods for broad band parallel applications in computational electromagnetics (CEM)," ICCFD2, 2nd Int. Conf. on CFD, Sydney Australia, July 2002

[2] Brio, M., Wu, C.C., "An upwind differencing scheme for the equations of ideal magneto-hydrodynamics," Journal of Computational Physics, Vol. 75, pg. 400-422, 1988

[3] Wu, C.C., "On MHD intermediate shocks," Geophysical Research Letters, Vol. 14, No. 6, pg. 668-671, June 1987

[4] Roe, P.L., Pike, J., "Efficient Construction and Utilization of Approximate Riemann Solutions," Computing Methods in Applied Sciences and Engineering, VI, Elsevier Science Publishers, INRIA, 1984.

[5] Sutton, G.W., Sherman, A., "Engineering Magnetohydrodynamics," McGraw-Hill Inc., 1965

[6] Atkins, H.L, Shu, C.W., "Quadrature-Free Implementation of Discontinuous Galerkin Method for Hyperbolic Equations", ICASE Report No. 96-51 (1996).

[7] Chakravarthy, S.R., "High Resolution Upwind Formulations for the Navier-Stokes Equations," VKI Lecture Series, 1988-05

[8] Cockburn, B., Lin, S.Y., Shu, C.W., "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One dimensional systems", Journal of Computational Physics, V84 (1989), pp. 90-113.

[9] Jiang, G.-S., Wu, C.-C., "A high-order WENO finite difference scheme for the equations of ideal magneto-hydrodynamics," Journal Comp. Phys., Vol. 150, pg. 561-594, 1999

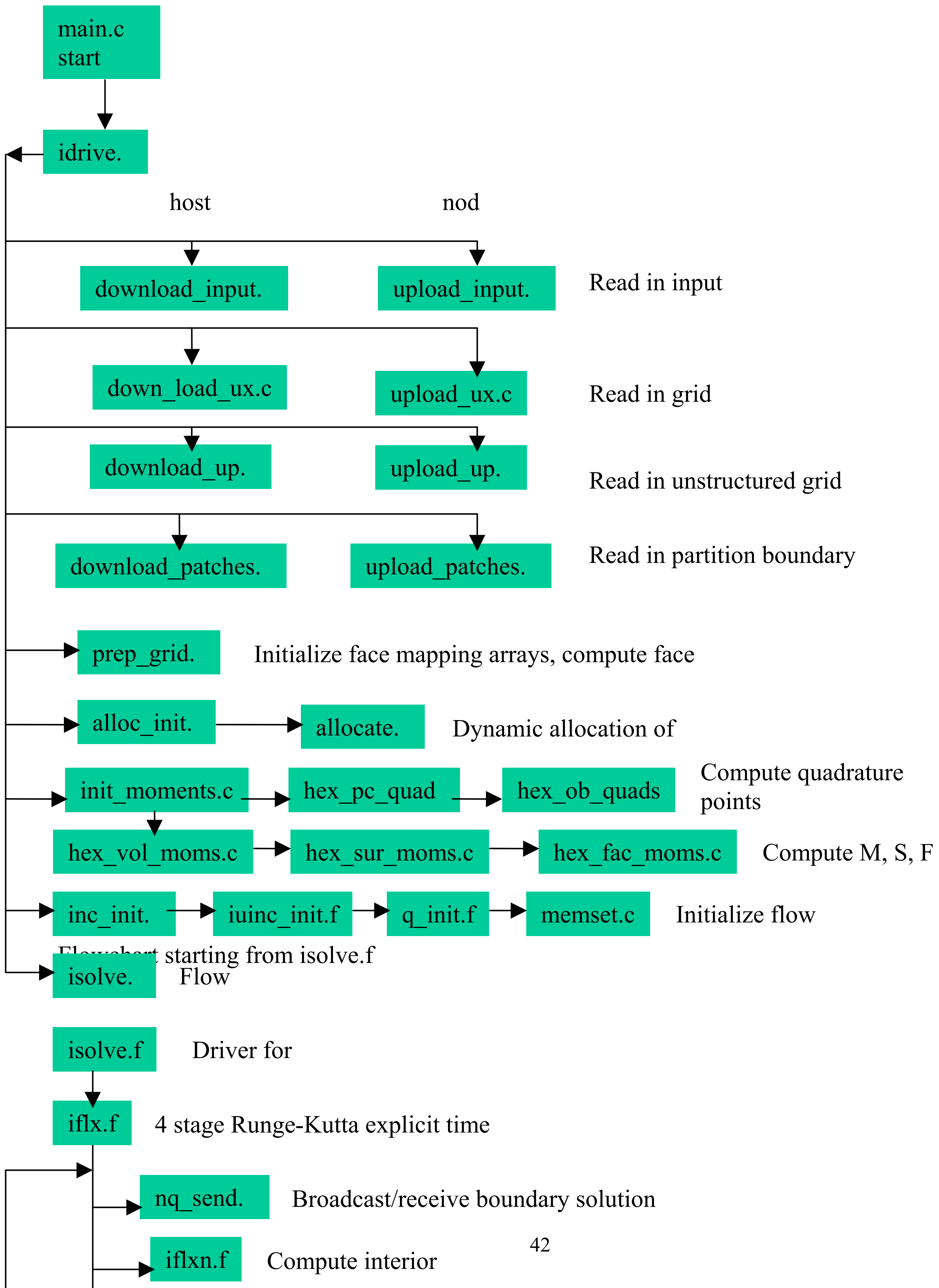
[10] Xu, K., "Gas-kinetic theory based flux splitting method for ideal magnetohydrodynamics," ICASE report 98-53, November 1998

[11] Munipalli, R., Shankar V., "Development of computational capabilities in real gas MHD simulations," AIAA paper 2001-0198, 39th AIAA Aerospace Sciences meeting, Reno, NV, January 2001

[12] Chakravarthy, S.R., Szema, K.Y., "Unified Nose-to-Tail Computational Method for Hypersonic Vehicle Applications", AIAA Paper No. 88-2564, 1988.

Appendix 1

Flowchart starting from main.c


main.c start
idrive.
host
nod
download_input.
upload_input.
Read in input
down_load_ux.c
upload_ux.c
Read in grid
download_up.
upload_up.
Read in unstructured grid
download_patches.
upload_patches.
Read in partition boundary
prep_grid.
Initialize face mapping arrays, compute face
alloc_init.
allocate.
Dynamic allocation of
init_moments.c
hex_pc_quad
hex_ob_quads
Compute quadrature points
hex_vol_moms.c
hex_sur_moms.c
hex_fac_moms.c
Compute M, S, F
inc_init.
iuinc_init.f
q_init.f
memset.c
Initialize flow
Flowchart starting from isolve.f
isolve.
Flow
isolve.f
Driver for
iflx.f
4 stage Runge-Kutta explicit time
nq_send.
Broadcast/receive boundary solution
iflxn.f
Compute interior